Fixed Verse Generation using Neural Word Embeddings

by

Arjun Magge

A Thesis Presented in Partial Fulfillment
of the Requirements for the Degree
Master of Science

Approved May 2016 by the
Graduate Supervisory Committee:

Violet R. Syrotiuk, Chair
Chitta Baral
Cynthia Hogue
Rida Bazzi

ARIZONA STATE UNIVERSITY

August 2016

ABSTRACT

For the past three decades, the design of an effective strategy for generating poetry that matches that of a human's creative capabilities and complexities has been an elusive goal in artificial intelligence (AI) and natural language generation (NLG) research, and among linguistic creativity researchers in particular. This thesis presents a novel approach to fixed verse poetry generation using neural word embeddings. During the course of generation, a two layered poetry classifier is developed. The first layer uses a lexicon based method to classify poems into types based on form and structure, and the second layer uses a supervised classification method to classify poems into subtypes based on content with an accuracy of 92%. The system then uses a two-layer neural network to generate poetry based on word similarities and word movements in a 50-dimensional vector space.

The verses generated by the system are evaluated using rhyme, rhythm, syllable counts and stress patterns. These computational features of language are considered for generating haikus, limericks and iambic pentameter verses. The generated poems are evaluated using a Turing test on both experts and non-experts. The user study finds that only 38% computer generated poems were correctly identified by non-experts while 65% of the computer generated poems were correctly identified by experts. Although the system does not pass the Turing test, the results from the Turing test suggest an improvement of over 17% when compared to previous methods which use Turing tests to evaluate poetry generators.

To family, friends,

and warm-hearted people of

the Himalayas.

ACKNOWLEDGMENTS

Writing this thesis has been a challenging and incredible journey. It would not have been possible without my mentors, colleagues, friends and family who have motivated me during my time as a graduate student. I am profoundly grateful to have Violet Syrotiuk, Cynthia Hogue, Chitta Baral, and Rida Bazzi on my thesis committee, all of whom are extraordinary scholars in their areas of research, and inspiring mentors. This thesis would not have been realized without their exceptional wisdom, belief, and patience.

I am particularly thankful to Violet Syrotiuk, for letting me run with the idea of generating poetry. Her invaluable guidance, constant support, and patience over the past year has been pivotal to this thesis. I am deeply indebted to Cynthia Hogue for her continuous encouragement, book recommendations, and for gracefully welcoming an engineer into her class. I am also very thankful to Rida Bazzi for his vital inputs and for helping me design, shape and plan the user study. I am especially grateful to Chitta Baral for helping me get started on natural language processing, artificial intelligence and machine learning, all of which form the pillars for this research.

Many thanks to Pablo Gervás, Hisar Manurang, Simon Colton and other researchers whose inspiring contributions in linguistic creativity has fueled this research. I owe a lot to the scientific community which provided me the tools, softwares and datasets used in my method. Special thanks to the anonymous participants of my user study, especially the students of Dr. Hogue's poetry workshop for serving as the expert group.

I thank Arizona State University for providing me the opportunities in the pursuit of this degree. And finally, I thank the late Eleanor Roosevelt, for the much-needed weekly reassurances that I must do the things I think I can not do.

TABLE OF CONTENTS

CHAPTER Page

CHAPTER Page

LIST OF TABLES

LIST OF FIGURES

Chapter 1

INTRODUCTION

Since the advent of computers, we have tried to employ them to automate mechanical and repetitive human tasks and have been largely successful. Over the years these tasks have ranged from mechanical jobs such as robots in a manufacturing industry [74] to cognitive tasks such as stock trading [68] and diagnosing diseases [47]. By defeating humans in memory-intensive and strategic games such as Chess [11], Jeopardy [31] and Go [93], computers over the years have demonstrated that the number of problems that appear to require human intelligence to solve is ever-receding.

1.1 Computational Creativity

Creativity has been considered an innate human capability that is inadequately understood from a computational standpoint, hence making its automation an interesting challenge. This void has led to the emergence of a branch of artificial intelligence (AI) research known as computational creativity.

Definition

Computational creativity has been largely defined as *the philosophy, science and engineering of computational systems which, by taking on particular 'responsibilities', exhibit behaviours that unbiased observers would deem to be creative* [20].

Responsibilities

The *responsibilities* in computational creativity have included generating cooking recipes [81], theorems [19, 21], board games [10], paintings [17, 61], music [29, 88, 72],

stories [104, 62], and poetry [27].

Many of these attempts to build creative systems have been documented by [20]. Paintings created by the robot AARON designed by Harold Cohen [61] have been exhibited and sold in art exhibitions. Artworks generated by the The Painting Fool software have been sold [17]. Theorems generated by the Hardy-Ramanujan (HR) discovery system have been published [16, 94]. Similarly, a board game invented by the Ludi system has been sold [10]. In music, the chorale harmonizations produced by the CHORAL system [29] could only be distinguished from those of J. S. Bach by experts and the Continuator jazz improvisation system has been used in performances with professional musicians [72].

1.2 Computational Linguistic Creativity

Linguistic creativity, a research area in linguistics on its own, is a sub-genre within computational creativity which deals with generation of text by computers in a fashion that can be deemed creative and meaningful.

1.2.1 Motivation

Creative writing and poetry are 'artistic' benchmarks that both AI research communities and natural language generation (NLG) communities would like to surpass in the years to come. NLG is a sub-field of both AI and natural language processing (NLP) whose objective is to generate understandable texts for humans given a particular message which needs to be conveyed to the user of the machine. The majority of the work in NLG has been in the discipline of text summarization typically used in generating concise descriptions of news contents based on space constraints [12, 102].

Apart from the AI, NLP and NLG challenges, computer assisted poetry generation may have multiple benefits. Fixed verse poetry in its various forms is popular

among the masses for its 'catchy' rhyme and meter which has a positive influence on humans' capabilities to memorize long passages of text. Some studies have shown that rhymes lead to better reading performance in children in addition to better phonological awareness [42]. The STANDUP generator, which is a punning riddle generator achieves positive results in children with complex communication needs [59] by. Some research has also shown that career professionals in medicine have an overall reduction in stress levels when they engage in creative arts such as writing and poetry [92]. We believe that an interactive poetry generation system would have the capability to assist humans, both young and old, to engage in the creative art by providing a starting point for generating ideas.

1.2.2 Challenges

Linguist Noam Chomsky states that the property of *Universal Grammar* resides in language (and the human capacity for it) which *"provides the means for expressing indefinitely many thoughts and for reacting appropriately in an indefinite range of new situations* [15]. This innate ability of understanding and using grammar in languages appears to be honed in children during their early stages of development [9] where they learn communication methods and develop linguistic capabilities which are later used to produce text which is creative, grammatically correct and semantically accurate. In comparison, programming computers to identify grammars and generate syntactically correct sentences has been considered an easier task. However, generating text which meet the semantic requirements demanded in poetry and prose is a difficult goal.

1.3 Poetry Generation

The 'art' of poetry and verse generation using computers has been of interest from as early as 1960 [6]. The *Stochastische Texte* [54] system used a grammatical

template to fit a set of sixteen subjects and sixteen predicates from Franz Kafka's work to generate a poem. Stochastic methods such as these have been used in many attempts at poetry generation since [13, 14, 106].

Poetry generated from strict constraints is considered a genre of its own that stems from the work of French mathematician Francois de Lionnais and writer Raymond Queneau named *Ouvroir de littérature potentielle* (OULIPO), or Workshop of Potential Literature in the 1960's [87]. There have been numerous attempts and techniques used since then to generate poetry in its accepted forms: free verse and fixed verse.

1.3.1 Free Verse

"The poet who writes "free" verse is like Robinson Crusoe on his desert island: he must do all his cooking, laundry and darning for himself. In a few exceptional cases, this manly independence produces something original and impressive, but more often the result is squalor dirty sheets on the unmade bed and empty bottles on the unswept floor."

W. H. Auden

Free verse poetry is free from rules such as the verse's meter, stress patterns and rhythm. The generation of free verse with computers can be as difficult as with humans given the free reign of meter and rhythm which in addition to the creator's poetic license may result in texts which may or may not be called a poem. Hence, for the purpose of this research we consider free verse poetry only as an input to understanding the verse structures and identifying the various challenges in tokenizing the verses and accurately identifying the Part-Of-Speech (POS) of the words themselves.

1.3.2 Fixed Verse

Fixed verse forms of poetry consists of rules in meter, rhythm and number of lines in a stanza. In this thesis we will focus specifically on three types of poetry which demonstrate all three characteristics of fixed verse: haikus, limericks and verses in iambic pentameters. Haikus are often called "one-breath poems" which contain a total of 17 syllables in 3 lines in a 5-7-5 pattern. Limericks are often spread over 5 lines and have a rhyming pattern of AABBA. Iambic pentameters are composed of alternate patterns of primary stresses and no stresses.

1.4 Objective

In this thesis we present a system that provides a novel approach to generating fixed verse poetry using neural word embeddings. In the process we develop a general purpose poetry classifier which can be used to classify poetry. To measure the success of our method we setup our hypotheses and design a user study that uses a Turing test to evaluate our poems. Results from the Turing test suggest an improvement of over 17% when compared to previous methods which use Turing tests to evaluate poetry generators.

The rest of the thesis is organized as follows. Chapter 2 provides a brief history and overview of related work in poetry generation methods. Chapter 3 introduces the system architecture and describes the components of poetry generation in detail. Chapter 4 establishes the hypotheses and also describes design of the user study for evaluating the poems generated to test the hypotheses. In chapter 5, we discuss the results of the user study and test the hypotheses. Chapter 6 lists some of the limitations the method and suggests future enhancements. Conclusions for the thesis are drawn in chapter 7.

Chapter 2

RELATED WORK

Poetry generation using computers has a long history which spans many decades. We describe each of the three broad categories of poetry generation and summarize the various methods used.

2.1 Families of Poetry Generators

Prior poetry generation methods broadly fall into three families: stochastic methods, multi-agent systems and miners.

2.1.1 Stochastic Methods

Stochastic methods rely on the syntax or grammar of the verses to substitute words/phrases in the text. Substitution is often performed by random selection of synonyms of words in the original text. Examples include the early works such as the *Stochastische Texte* [54] system and Jim Carpenter's Electronic Text Composition (ETC) project [13] among others [14, 106, 1].

2.1.2 Multi-agent Systems

The second family consists of multi-agent systems which try to mimic the process typically used by humans to generate poetry. Some of these methods are composed of different modules which perform individual tasks that generate the lines, review them, modify them, and repeat the process for a few times to arrive at a generated poem [22, 57, 34]. The review system in such methods is enforced by establishing rule based programs for evaluating the poetry such as affect, imagery, and meaningfulness

There is pleasure to

be had here, in flares of spice

that revive and warm.

Figure 2.1: A haiku mined by the Times Haiku bot

to maintain the aesthetics of the poem. Additionally, there are a few methods which try to generate serendipitous text and/or generate similes as they are considered important elements in poetry [23, 18, 71].

2.1.3 Miners

The family of poetry generators include automated poetry bots. These bots mine social media or blogs on the internet looking for content which fit its constraints. For instance, Times Haiku bot [44] mines the New York Times articles for content which roughly fit the Haiku format and the best among them are selected for publishing on the blog.

Similarly, the Pentametron twitter bot [8] looks for rhyming couplets in tweets that fit the syntax of an iambic pentameter. It then publishes fourteen such tweets to form a Shakespearean sonnet. Examples for poems mined by the Times Haiku bot and Pentametron are shown in Figure 2.1 and 2.2.

Poems generated by Pentametron are not inherently designed to be meaningful because every line in the poem is an individual tweet about mutually unrelated topics. However, the result of the Times Haiku bot is likely to carry more meaning as each poem is extracted from a sentence.

For the purpose of this thesis, we will omit the mining bots and works based on the OULIPO method [67] as the method of generation. We focus our attention on the recent methods used to generate poetry ever since statistical NLP and AI techniques

I'm going swimming after school #hooray

I wanna hear a special song today :) !

Last project presentation of the year!!!!

Miami Sunset Drive A. normal clear :)

Good music always helps the morning squat!!!!

McDonalds breAkfast always hit the spot

do you remember ? that october night ..

Alright alright alright alright alright

I taught y'all bitches half the shit y'all know.

Why pablo hating on Hondurans though ?

I wonder who the Broncos gonna pick?

I gotta get myself a swagger stick

By Changing Nothing, Nothing changes. #Right?

Why everybody eagle fans tonight

Figure 2.2: A sonnet titled 'Why everybody eagle fans tonight' mined by Pentametron

have been introduced into the field. In the following sections, we list and summarize major works accomplished in the area of poetry generation based on their method used for generation, source of text used for form and content, and evaluation methods.

There has been a shift in the core methodologies used in poetry generation methods, from word substitution tricks in the second half of 20th century to statistical NLP and AI techniques in late 1990's. Ray Kurzweil's Cybernetic Poet (RKCP) [51] used a proprietary algorithm which took a collection of poems from a poet or a com-

bination of poets to generate a language model similar to Markov models. It uses this language model to generate text matching the rules of the type of poem selected. An RKCP poem is shown in Figure 2.3.

Ages and pink in Sex,

Offspring of the

voices of all my Body.

Figure 2.3: A haiku titled 'And Pink In Sex' written by Ray Kurzweil's Cybernetic Poet after reading poems by Walt Whitman

Pablo Gervás developed the Wishful Automatic Spanish Poetry (WASP) [34] and ASPID [41] methods for generating Spanish verse based on a similar approach that used predefined rules for a poem encoded in NASA's CLIPS based rule system [89]. An extension of the approach in ASPERA [36, 35, 37] used Case Based Reasoning (CBR) approach where the rules were extracted from the structure of the poem that was provided by the user. The word substitutions for both methods relied on random selections based on part-of-speech (POS)-tagged words from prose provided by the user.

The CBR approach is further extended in COLIBRI (Cases and Ontology Libraries Integration for Building Reasoning Infrastructures) [27] where the authors describe a knowledge representation ontology for the target poem. The methods uses Problem Solving Methods (PSM) to retrieve solutions for each case in the ontology. The substitution method uses an iterative process where each round of substitution is followed by an evaluation of the cases. After multiple iterations the verse with the highest evaluated score is selected as the output.

Hisar Manurung's McGONAGALL system [57] was one of the first to use a semantic representation to define the target poem's structure. Based on his earlier work on

generating rhythm patterned text [58], he used a stochastic hill climbing search for substituting words. The generated poem would then be checked as satisfying three properties namely, grammaticality (syntactical structure), meaningfulness (semantical structure), and poeticness (presence of figurative language and desired rhythmic patterns) in an incremental approach to arrive at the final poem. McGONAGALL's representation scheme for meter and semantics used a flat Lexicalized Tree Adjoining Grammar (LTAG).

In the statistical machine translation (SMT) approach used by [48], the system takes the first sentence as an input from the user and generates a list of second sentences based on the phrase-based SMT decoder. Linguistic constraints for Chinese couplets are checked and violating candidates are removed from the list. Finally, a support vector machines (SVM) based ranking method is used to produce the top-ranked output.

The SMT approach used a combination of human and machine evaluation on a set of 100 generated couplets to determine the effectiveness of the approach and demonstrate the feasibility of machine evaluations. The human evaluation was performed on a binary scale: 0 for reject and 1 for accept and machine evaluations used BLEU [75] scores that are commonly used in evaluation of SMT approaches. The comparison suggested a good correlation between both forms of evaluations.

Other methods include the adaptation of a writer's workshop [22] by Corneli et al. for development of a poet/writer in a social context using the cycle of presentation, listening, feedback, questions, and reflections. Furthermore, they develop a model to identify and evaluate serendipity in a given system and model it for poetry generation in a multi-agent system [23].

Vector Models

The Vector Space Model (VSM) was first introduced in 1975 [91] for indexing documents and evaluating document similarity in information retrieval systems [90]. Although vector spaces can be used in many ways such as gene sequencing [5] and recommendation engines in social media [80], we describe the method in reference to recent advancements in the field of word vectors. [26, 63, 40, 52]. The proposed model in this thesis uses *neural word embeddings*, which is a representation of words as vectors, derived using training methods inspired from neural-network language modeling [7, 64, 66, 53].

Words in a text can be imagined as discrete states where vectors can be calculated using transitional probabilities between those states, i.e., the likelihood of co-occurrence. The vector space is built by processing a large amount of text. The vector values for each word is calculated across fixed number of dimensions based on its neighboring words which form its context. Thus, vectors are distributed numerical representations of word contexts and it can be built without human intervention. Each word in the vector space can be imagined to be a point in the vector space. The vector representation of a word in the model contains floating point values between 0 and 1 for each dimension.

The vector model can be built with deep-learning (neural-network) tools using models such as skip-gram which is used in *word2vec* [26] and *GloVe* [79] among others. The models have the potential to guess the meaning of the words based on previous occurrences and associations. For instance, the *semantic similarity* between two words is a measure given by the proximity of the two words in the vector space. It can be calculated using the cosine similarity between the two vectors, i.e., angle between the vectors across all dimensions.

The applications of this similarity measure in word vectors can be demonstrated using analogy problems. For instance, consider the problem of finding the solution of the analogy 'man : woman :: king : ?'. Here, the model can calculate the cosine similarity between 'man' and 'woman' and search the vector space for words which have similar values with respect to the word 'king'. The word that has the closest proportional value in the vector space is found to be 'queen' which is closely followed by 'princess'. Other interesting results as documented in [26] include:

China : Taiwan :: Russia: [Ukraine, Moscow, Moldova, Armenia]

building : architect :: software : [programmer, SecurityCenter, WinPcap]

New York Times : Sulzberger :: Fox : [Murdoch, Chernin, Bancroft, Ailes]

Vector models have been used previously to generate poetry. The VSM method used by Wong et. al. [105] to generate haiku relies on semantic similarity between sentences. The semantic similarity is calculated using the same method as described above. A seed word is initially used to search for sentence fragments from blogs. Vectors values are calculated between combinations of word-pairs extracted from each sentence and the collection of lines whose scores lie closest to 1 produce the output.

In contrast, the system proposed in this thesis tackles a larger problem where sentence fragments are not used, and the poem is built from scratch by replacing every word in the original template. The resulting poems have word sequences which are highly unlikely compared to poems built from sentence fragments.

In the next chapter, we introduce this new system which uses both multi-agent and stochastic approaches to generate poetry. The process begins by mining existing works of poetry from social media and poetry websites. The poems thus obtained are used to build a vector space using a skip-gram model for neural word embeddings. The poems are further used to create a template based on POS-tags for stochastic replacement of words based on word similarity and word movement measures.

Chapter 3

METHOD

We present a novel multi-agent system called *AutoPoe* which uses neural word embeddings to generate and evaluate poetry. The idea behind the proposed system relies on knowledge that can be built from a large number of human composed poems. We design the system to build a massive repository of poems, which is used not only to construct a reusable syntactic structure but also build the collective vocabulary of the system. The generation of every poem is triggered by a user supplied cue word. This word is considered to be the theme around which the generated poem should be based.

For every user provided theme, the system randomly selects an arbitrary number of poem templates from the repository. From each of these poem templates, we generate a new poem surrounding the new theme. All generated poems for the theme are evaluated by the system using semantic similarity measures between the theme and the generated verse. Among the evaluated verses, 10 are selected as the output. We provide a brief overview of the system architecture in the following section. Each step is described in detail along with its implementation in the subsequent sections.

3.1 Architecture Overview

We show *AutoPoe*'s system architecture in Figure 3.1. At the core of our proposed method for fixed verse generation, lies three major steps: knowledge acquisition, verse generation and evaluation. Unlike the steps of verse generation and evaluation which are triggered by user supplied theme, knowledge acquisition is a continuous process that runs independently.

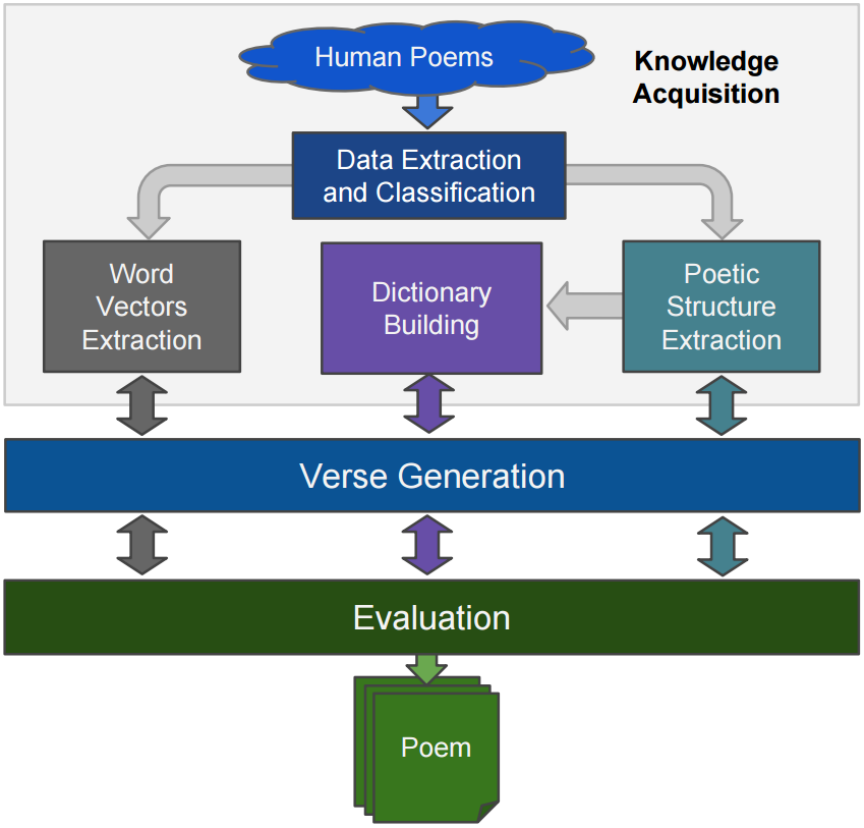

Figure 3.1: *AutoPoe*'s System Architecture

Knowledge Acquisition

The step of knowledge acquisition involves building the vocabulary, the grammatical structures in the language, semantic word associations, and the constraints for various types of fixed verse.

The components of collective knowledge described above are built from four subprocesses in the knowledge acquisition step as shown in 3.1. Data extraction first collects poems published in social media and digital poetry websites. The poems are then classified into types of fixed verse poetry based on the structural constraints and contents of the poetry. Undesired forms of poetry are discarded at this stage. These undesired forms include non-English poems and poems from an earlier era such as

16th-19th century where the vocabulary is found to be different from modern English poetry.

The classified poems along with their types and machine annotated texts are stored in the poetic structure repository which forms the basic templates for generation. Each individual word along with its usage detail is added to the dictionary to build the vocabulary of the system. Irrespective of the type of the accepted poems, the verses from the poems are used for building a word vector model which aid in semantic word associations. The the word vectors for the verses are built upon pre-trained word vectors generated from Wikipedia articles to include a larger vocabulary.

Verse Generation

The verse generation step is triggered by a user provided theme and type of fixed verse. The output poems from the system are expected to be formed around this theme and type of fixed verse. The verse generation step starts with a list of poems obtained from the poetic structure repository that belong to the desired type. Each poem goes through a process of substitution based on semantic similarities to the desired theme and to the original poem to generate many candidate poems. The potential substitutions are picked from the system's vocabulary, i.e., the dictionary, while semantic similarities are calculated using the word vector model.

Evaluation

For each of the candidate poems generated, a series of evaluations are performed to eliminate poems which do not conform to the desired constraints. These constraints include repeated use of words within the same poem and violation of the structural constraints of the type of poetry among others. The final step involves assigning a score the poem based on semantic relatedness of words within the poem. The

semantic similarities in the verse generation step and the semantic relatedness in the evaluation step are calculated using word vectors. Poems from a given range of scores are selected as the output and presented to the user.

3.2 Knowledge Acquisition

The process flow and components of the knowledge acquisition step along with their implementations have been illustrated in Figure 3.2. It consists of four sub-processes which are now described in detail.

3.2.1 Data Extraction and Classification

Data Extraction

The success of *AutoPoe*'s generation method largely relies on the availability of a large number of poems. While a publicly available large corpora comprising of poetry is unavailable, there are public websites dedicated to poetry like Poetry Foundation [83] that contains a wide range of fixed and free verse poetry by published poets and more. Data from such websites tend to be static due to a low number of additions per day. However, we found that many poems written by amateur poets can be found on social media. Twitter is a microblogging site which is actively used by over 320 million users [98] and we found that up to 10,000 short poems are published every day. Thus, to increase the number of poems for building the repository we compromise on both length of the poem and quality of the poem.

We used around 14,000 obtained from Poetry Foundation which contain a wide range of fixed and free verse poetry. We build a system which continually retrieves poetic content from Twitter for short form poetry. For twitter, we use its application program interface (API) to monitor and search for hashtags which are often used with poetry such as #haiku, #micropoetry, #3lines, #mpy and #5lines for shorter

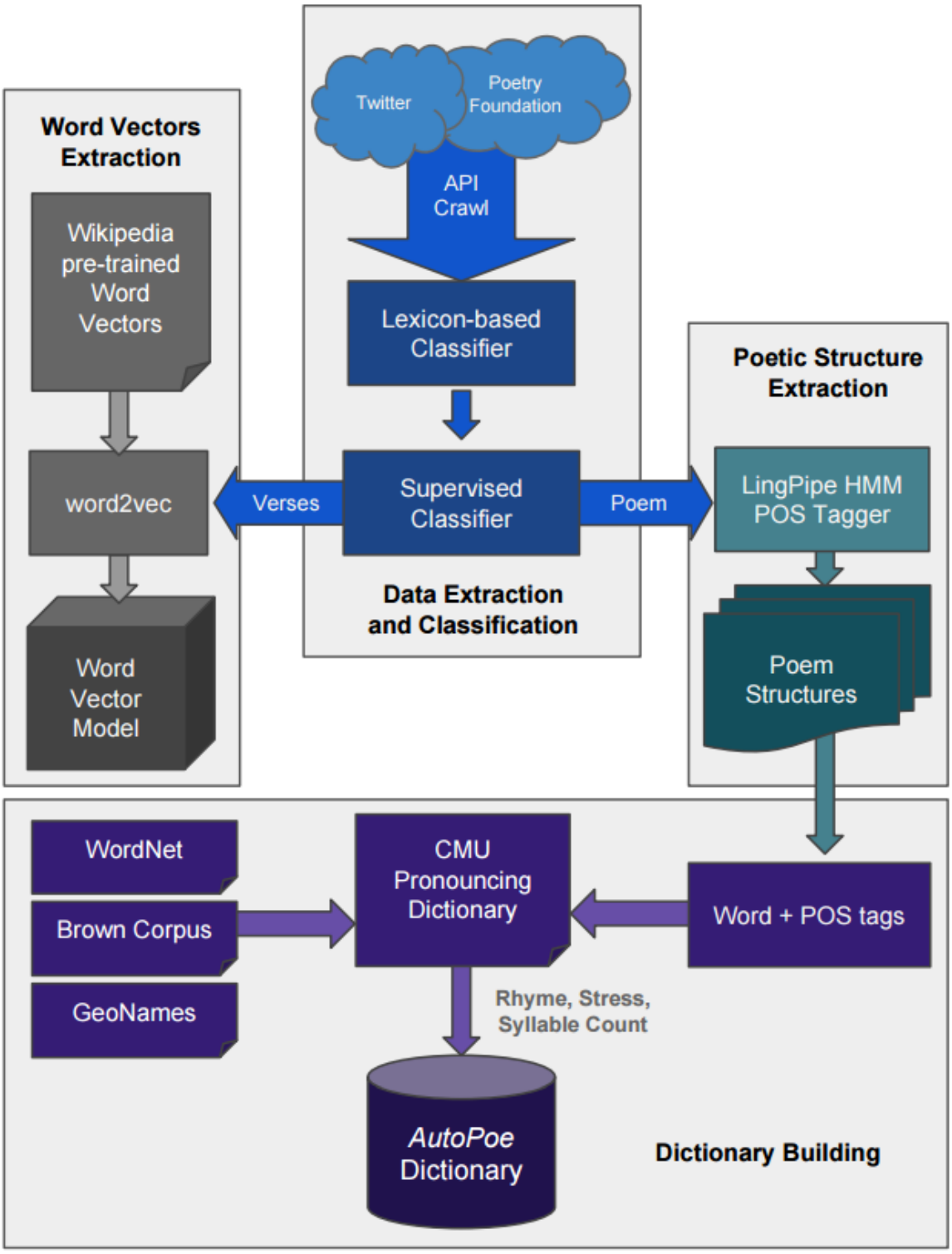

Figure 3.2: Knowledge Acquisition

poems like haiku and tanka which fit into the 140 character limit in Twitter. Using these five hashtags, we employ a social media mining technique called association rule mining [2] to search and expand the hashtag set. We were able to expand our set to 53 hashtags which provide can be used to track more than 10,000 poems published on twitter every day. The final set of hashtags that are monitored using the twitter API have been included in Appendix E.

Classification

We employ two techniques for classifying poems. The first step involves classifying poetry into fixed verse or free verse poetry. This involved processing the number of lines in the poetry along with its rhyming scheme, syllable counts and stress patterns to determine the type of fixed verse poetry. For instance, if the verse contained three lines with five syllables in the first line, seven in the second and five in the third, we classify it as haiku. Similarly, if the verse contained five lines where the rhyme pattern of the verse is A-A-B-B-A, we classify it as a limerick. If the poem matches none of the fixed verse constraints, then we classify it as free verse poetry.

The second step of classification is a non-trivial problem where the poetry type is determined by the contents of the poem. Take for instance, classical poetry and contemporary poetry. Some words such as *'tis', 'hath', 'thou'*, and *'thy'* which were commonly used in classical poetry have failed to continue in existence within contemporary vocabulary. Another example includes haikus and senryus, both of which have a total of 17 syllables spread across 3 lines. The Haiku Society of America calls haiku a *"poem in which Nature is linked to human nature"* while topics in senryus are *"specifically about human nature and human relationships and is often humorous"* [99].

We use a supervised classifier which can be used for classification tasks based on

the content of the poem. In our system, we use it to distinguish classical and modern poetry. We employ Weka's [43] Support Vector Machine (SVM) classifier using the Sequential Minimal Optimization (SMO) algorithm for optimization [82, 49, 46]. We use generic features of the language that can be easily extracted to avoid over-fitting the classifier model. We do this to maintain portability of the classifier when required in other sections of the pipeline or on a different set of class labels.

We train the classifier to include six individual features on two different datasets. The first three features include word sequences from the poem that are in a bag-of-words model [45]. The final feature is a binary value indicating the overall sentiment in the poem.

1. **Unigrams**

   We tokenize the sentences from the entire poem in a space separated sequence and include them in a bag-of-words model. Before adding the words, we remove frequent words such as articles and conjunctions (commonly known as stop-words) which do not value to the feature. For instance, the unigram feature for the text *"I was angry with my friend;"* would consist *"angry friend"*.

2. **Bigrams**

   Similar to unigrams, we tokenize the sentences in the poem into bigrams i.e. word pairs, where each pair of words is separated by an underscore symbol. When a line contains only one word which is not a stop-word, the word is ignored as it does not form a pair. Such cases are common occurrences in poetry, hence we do not remove stop-words in the bigram feature. For instance, if the sentence was *"I was angry with my friend;"*, we tokenize it into *I_was was_angry angry_with with_my my_friend*. Compared to unigrams, the number of unique bigrams are higher and hence memory intensive. However, the bigram

feature improves the classification accuracy in our system by 3%.

3. **POS-tags**

   In addition to unigrams and bigrams, we also add tokenized word_POS-tag sequence for including the broad sense in which the words occur. We use a maximum entropy tagger from the Stanford CoreNLP toolkit [56] to tag every word with one of the 36 possible PENN Treebank POS-tags [60]. We find that this feature marginally improves the overall classification by 1%. For the previously described text, we add the following set of tokens for this feature *I_PRP was_VBD angry_JJ with_IN my_PRP$ friend/NN*. Here, PRP is a personal pronoun, VBD is a verb in the past tense, JJ is an adjective, IN is a preposition/subordinating conjunction and NN is a singular noun.

4. **free association norms**

   Free associations are obtained by starting with a list of cue words for which responses (associations) need to be recorded. Each cue word is presented to a set of human subjects, who provide one-word responses known as targets. The cue-target are further ranked by frequency. Free association has been reliably used in for measuring word associations since 1964 [73, 24, 69]. This feature was found to improve the classification accuracy by 5%.

   We use University of South Florida's Free Association Norms database [70] for its large collection of 72,000 words pairs. From this database we compile two hash-maps for cue-targets (1:n) entries and target-cues (1:n) entries. For every given word in the poem that is not a stop-word, we obtain the top-10 most frequent words for which the given word was a target and top-10 targets when the given word was used as a cue. For the given example, *angry* is associated with the words *mad, upset, violent, fight, aggressive, etc.* and *friend* would

be associated with *pal, buddy, foe, advice, trust, etc.* The free associations often include commonly used classifier features such as synonyms, antonyms and collocated words. Hence, we do not use those features explicitly.

5. **Sentiment polarity**

   The differences in various forms of poetry often lies in its contents. Compared to haikus, senryus tend to express an overall positive sentiment due to the presence of humor. These distinctions appear in collections of poems published by poets, hence we employ sentiment polarity scores for each poem. We use SentiWordNet [30] which contains sentiment scores for more than 117,000 entries. Each entry in SentiWordNet contains a word and its POS-tag along with its positive sentiment score, S_P and negative sentiment score S_N. For each word in the poem that is not a stop-word, we calculate its normalized sentiment score S_p using the Equation 3.1 shown below.

$$S_p = \frac{\sum_{i=1}^{n} S_{Pi} - S_{Ni}}{n} \tag{3.1}$$

   where,

   n is the number of non-stop-words in the poem,

   S_{Pi} is the positive sentiment for word i, and

   S_{Ni} is the negative sentiment for word i

For this thesis we classify the poems retrieved from Poetry Foundation into classic and modern classes for the sole purpose of generating poetry belonging to the current era. The training set for the classification contained 500 randomly selected poems from the Poetry Foundation poem collection. Each poem was annotated as *classic* or *modern* based on the poets and classified using our SVM classifier. The results from the classification task have been tabulated in Table 3.1.

Classification between	Precision	Recall	F-measure
Classical and Modern	0.931	0.912	0.921
Haiku and Senryu	0.647	0.682	0.662

Table 3.1: Supervised Classification Results

In binary classification methods, Precision is the ratio of the true positives to the sum of true positives and false positives i.e. *TP/(TP+FP)*. Recall is the ratio of true positives to the sum of true positives and false negatives i.e. *TP/(TP+FN)*. The F-measure is the harmonic mean of Precision and Recall. The classification of classic and modern poetry yielded a precision of 93.1% and recall of 91.2%. The overall classification score indicated by the F-measure was found to be 92.1%. The errors observed in the classification were mainly due to incorrect annotations as some modern poets wrote poems in classical style.

In comparison, classification of three line poems into haiku or senryu had an overall classification score of 66%. This classification accuracy in this case is low for two reasons. Firstly, we believe the annotations were incorrect in some poems as the classes were based on use of hashtags, and second because even human beings have difficulty telling haikus and senryus from each other.

Our classification yielded a total of 11,038 poems from Poetry Foundation belonging to the modern class. Once classified, the modern poems are saved, from where both the word vectors and poetic structures are built.

3.2.2 Word Vectors Extraction

To discover semantic associations and similarities among words we build a neural network consisting of two-layers that processes text. It takes words from a given

corpus and turns it into a set of vectors in that corpus. With such an vector model, we can group words which are semantically similar using cosine similarity [96]. For this task we use the *word2vec* implementation by Deeplearning4j [25].

We build our vector model on top of a pre-trained word vector model from Wikipedia articles and the Gigaword corpus [39] that was trained on 6 billion words across 50 dimensions [84]. The verses from the classified poems are processed and tokenized prior to being fed into the *word2vec* engine. We load the vector model from Wikipedia and update the vectors with the processed verses. We update the word vector model with poems irrespective of the form of the poetry, i.e., free verse or fixed verse because at this stage we are interested in the content of the poems rather than their form.

Although we receive a continuous supply of short-length poetry from twitter, we only update the vector model once every 24 hours as building/updating the vector model is computationally expensive. We use the skip-gram model [63, 40] in word2vec where we use a word to predict a target context. The traditional method used a continuous-bag-of-words (CBOW) model where the context was used to predict the target word. The difference in the two models is illustrated in Figure 3.3. During the training process of skip-gram model, the vector values assigned to an input word is constantly evaluated in the projection step to verify if can be used to predict that word's context. When it can not predict it accurately, the vector values are adjusted iteratively until the prediction is successful. In the CBOW model, the process is reversed where the context is used to predict the word. We use the skip-gram model as it is more efficient than the CBOW model for detecting semantic similarities [63]. The word vector model built in this phase is one of the crucial components used in our verse generation step.

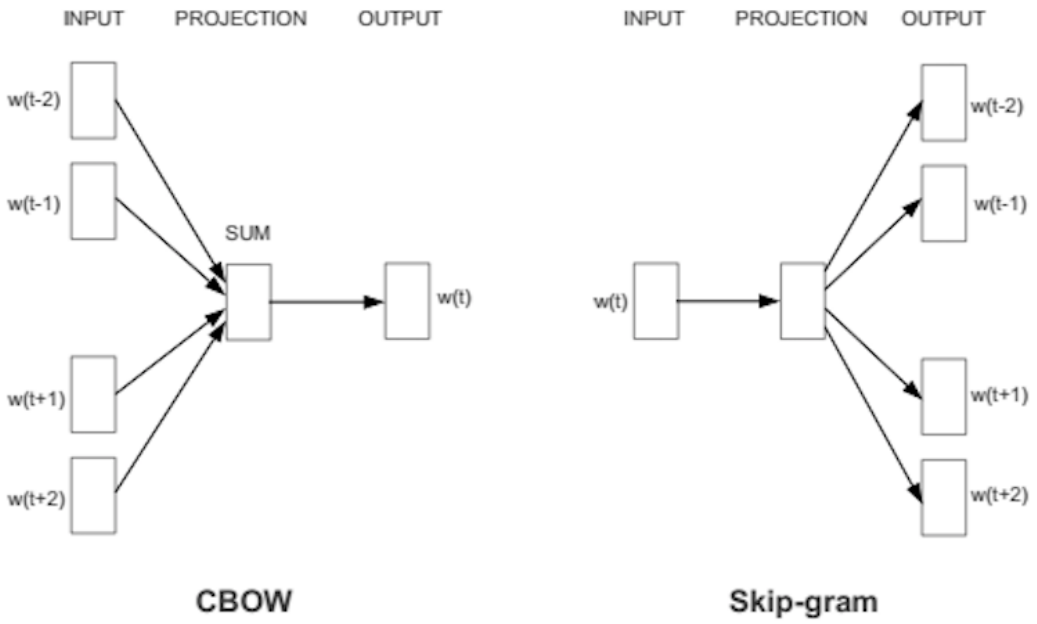

Figure 3.3: Difference in the Bag-of-words and Skip-gram Model [26]

3.2.3 Poetic Structure Extraction

We create the poetic structures based on the syntax used in the verses of the poem. The syntax here collectively refers to the templates generated from POS-tagging lines of the poem and the inherent verse constraints in the desired poem.

Brown Corpus

For POS-tagging the lines in the poem we need to train the tagger with a human annotated corpus consisting of a word and its part-of-speech tag. For natural languages, one of the highly used human annotated corpora is composed of the Wall Street Journal articles [76] which uses the PENN treebank annotation scheme consisting of 36 possible tags [60].

However, we select an older human annotated corpus known as the Brown corpus [32] because it uses a much larger tag-set consisting of 85 tags in addition to:

1. **Corpus size**

   The Brown corpus contained more than 1 million annotated words from texts. By comparison other freely available annotated corpora such as Wall Street Journal (WSJ) contain much fewer words.

2. **Corpus diversity**

   The texts that make up the Brown corpus range from 15 different topics that range from press journals to biographies and fiction. In comparison the highly used WSJ annotated corpus contains articles only from the journal which is not ideal for our POS-tagger because we intend to use it for tagging verses in poetry.

3. **Corpus tagset**

   Fundamentally, with the increase in number of tags the overall tagging accuracy decreases. However, for purposes of verse generation it is important to have a higher number of categories for appropriate usage of vocabulary which conform to the syntactical constraints in the verses to reduce ambiguity.

LingPipe POS-tagger

We train a unidirectional Hidden Markov Model (HMM) [85] POS-tagger on the Brown corpus using the implementation provided by a tagger tool called LingPipe [3]. We use LingPipe for its flexible options for building the POS-tagger in addition to its seamless integration with a Brown corpus. Before we train, we eliminate some of the extraneous tags in the Brown corpus such as foreign words denoted by the prefix *fw* and superficial details such as title presence, hyphenated words and cited words denoted by *tl, hl* and *nc*. We train the POS-tagger on the Brown corpus with the Lambda factor interpolation parameter for smoothing HMM emissions set to 8

which has been found to be reasonably accurate for natural languages [107].

The output generated from this step includes a large quantity of poetic structures, i.e., templates based on which new poems will be generated. These templates also include the type of poetry for which the templates are suitable. The poetic structures generated are also used to build the vocabulary of the system in collaboration with other knowledge.

3.2.4 Dictionary Building

To improve the quality of the poetry the system needs to have a large vocabulary. It takes special skill to find new words which fit the constraints of the verse. However, with a large vocabulary, machines can perform the art of suggestion quite easily, given the grammar. We build a custom dictionary from various sources due to the lack of a single multi-purpose dictionary. The first basic vocabulary that we build, uses the Brown corpus' annotated text. It is tokenized using space and sentence delimiters before being written to the index. These entries carry a higher priority over the other words. Each word in the corpus is an entry in the dictionary. Each entry contains the word, its annotated tag, rhyming scheme, stress pattern and syllable count. Each entry also contains the term frequency obtained from the poetry corpus and Brown corpus. Each word is also checked if it is a proper noun and available in the Geonames database of 15,000 cities [103] and if so, is added to the index with a particular flag set.

For storing the dictionary, we design the system to store each record in a Apache Lucene's index [4]. The poetic structure dictionary outputs template poems along with their POS-tags. We begin by indexing the words learnt from the Brown corpus, sorted by term frequency. We find that synonym sets (synsets) using WordNet [65] provides an easier way to extend the vocabulary. For rhyme, meter and stress cal-

culations, CMU pronouncing dictionary [101] is used. This pronouncing dictionary contains phonetic information associated with every word such as stress, rhyming schemes and syllable counts. These aid in stochastic methods of replacing templates derived from the poem. In the end, our *AutoPoe* dictionary consists of over 70,000 individual words along with their POS-tags.

3.3 Verse Generation

Verse generation requires three necessary elements from the knowledge acquisition step to function: the *AutoPoe* dictionary, the poem structure repository and word vector model. An illustration of the components involved in verse generation is shown in Figure 3.4.

The generation step consists of searching the *AutoPoe* dictionary for words that fit the grammatical and poetic constraints and evaluating the results to find a candidate word for replacement. The step is triggered by the user who specifies the type of fixed verse poetry desired and provides a word as the theme for the output poem. Based on the type of poetry, 10 templates from the poem structure repository are selected at random. Each template goes through the following steps to generate the poem.

3.3.1 Stochastic Search

For every word position in the poem template, we retrieve the original word in the template, its POS-tag, and the constraints. We create a query based on the constraints and retrieve all results from the indexed dictionary that match the query. Depending on the type of the poem, the line number of the poem and the position of the word, the query would contain additional filters for syllable count, rhyme pattern and stress pattern.

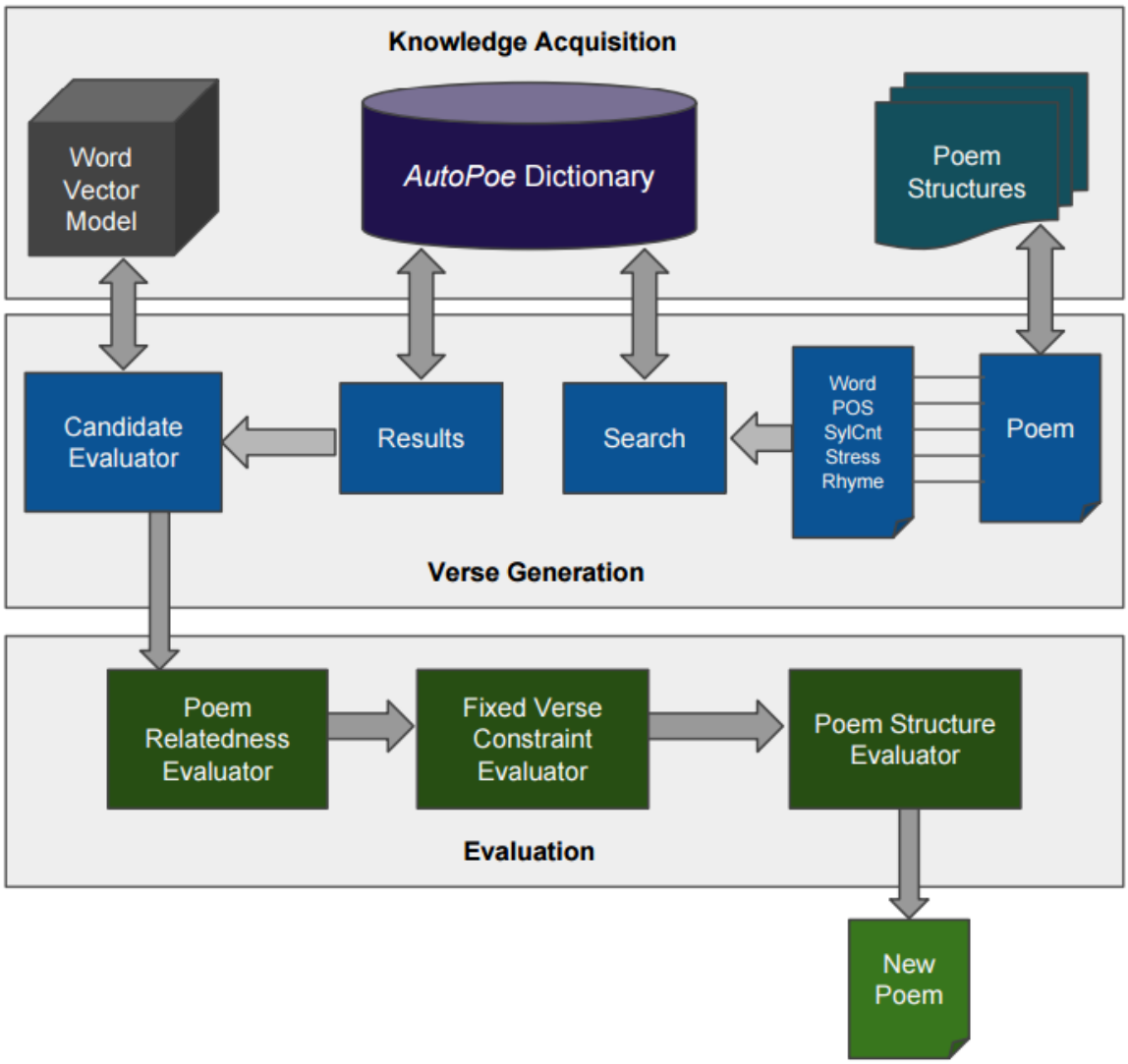

Figure 3.4: Verse Generation and Evaluation

3.3.2 Candidate Words

All results are candidates for replacement of the original word in the resulting poem. We choose the replacement word by calculating the replacement score R_w which maximizes the cosine similarities between the candidate word, original word and the theme, see Equation 3.2. We introduce a similarity constant α which regulates the distance between the theme and the original word. For this thesis, we generate poetry with the value of α kept to 0.75.

$$R_w = \max_{i=1}^{n} \; [\; \alpha \; C_s(w, c_i) + (1 - \alpha) \; C_s(c_i, t) \;] \tag{3.2}$$

where,

n is the number of candidates,

w is the original word that needs to be replaced,

c_i is the i^{th} candidate word,

C_s(a,b) is the cosine similarity between the two words a and b, and

t is the desired theme for the poem

For multiple themes, we take the average of R_w for each theme. Increasing the number of themes inccreases the computational complexity of the system. Here we use the cosine similarity C_s for the pair of words and not the cosine distance C_d. The relationship between the two is shown in Equation 3.3.

$$C_s = 1 - C_d \tag{3.3}$$

Hence, maximizing the cosine similarities would equate to minimizing the word distances. Word distance from word vectors have been used previously to find document similarities [100, 52].

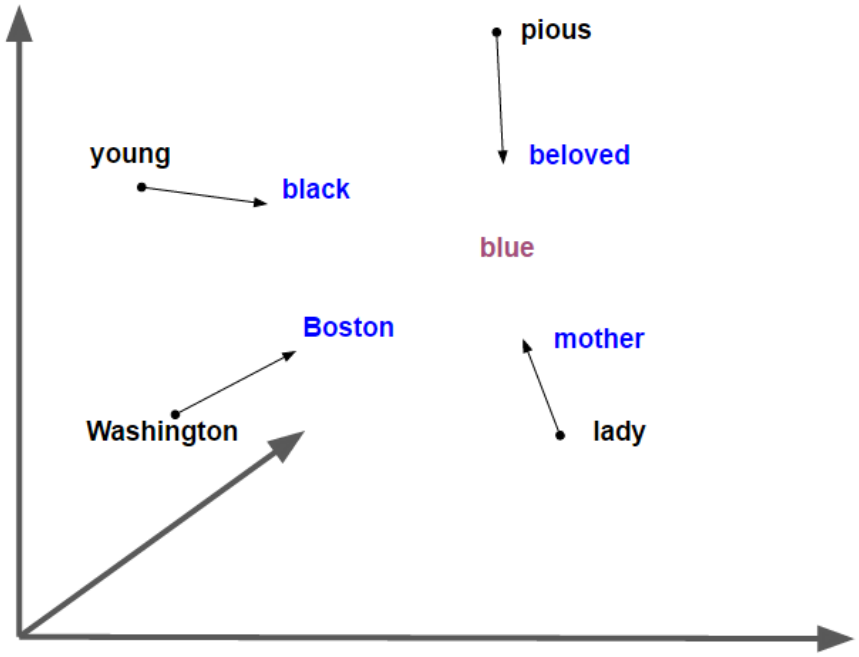

Figure 3.5: 2d Visualization of Word Movements Towards a Theme of 'Blue'

To illustrate with an example, take the first line of a limerick *"A pious young lady of Washington"*. Table 3.2 shows the top 5 results for the four words ranked by their replacement scores R_w. The arbitrary theme chosen for the poem is *blue* and α is set to 0.25 to show the similarity to the original poem. Figure 3.5 shows a 2-D representation of a vector space to demonstrate movement of words towards a theme of blue.

The associations made by word vectors relate how words are associated with each other. For instance, all alternatives for Washington were places in United States. Had this been a location in the United Kingdom (UK), places in its proximity would be ranked very high in similarity. One of the adjective alternatives to "young" having proximity to the theme of "blue" is "black". Hence, word vectors prove to be a very strong tool in semantic similarity measures.

Among the replacement words which are sorted by the scores calculated in Equa-

Original Word (w)	POS of original word w	Replacement Word (c_i)	Replacement Score (R_w)
		faithful	0.5359
		roman	0.5069
pious	adjective (jj)	virtuous	0.5042
		beloved	0.5032
		religious	0.5014
		black	0.7334
		fellow	0.7037
young	adjective (jj)	female	0.6925
		american	0.6896
		little	0.6684
		queen	0.7100
		mother	0.6329
lady	singular noun (nn)	daughter	0.6288
		sister	0.6198
		green	0.6146
		york	0.7355
		chicago	0.6552
washington	proper noun (np)	clinton	0.6539
		boston	0.6286
		florida	0.6259

Table 3.2: Replacement Words and Scores with $\alpha = 0.25$ for the Example *"A pious young lady of Washington"*

tion 3.2, we pick a fixed number of potential replacement words denoted by H. The output of the generation step is a 2-dimensional matrix of replacement words, where the number of rows indicate the number of words in the poem and the number of columns represent the number of replacement words for each position (H) to be evaluated.

3.4 Evaluation

The tasks performed in the evaluation step are shown in Figure 3.4. In this step we create combinations of words from the matrix to generate multiple poems and rank them by their internal relatedness score. Every poem is checked against the poetic constraints and the original word structure. The output of this step is a list of poems with their candidate scores from which we choose the final poem.

3.4.1 Poem Relatedness Evaluator

With the two-dimensional matrix of words, we generate multiple combinations of poems by randomly selecting a column for each row. For each poem, we calculate the internal similarity S_I of words in the poem using the Equation 3.4.

$$S_I = \frac{\sum_{i,j}^{n} C_s(w_i, w_j)}{N} \tag{3.4}$$

where,

n is the number of positions that need replacement i.e. H,

w_i and w_j are the i^{th} and j^{th} words in H,

$C_s(w_i, w_j)$ is the cosine similarity between the two words w_i and w_j, and

N is the number of unique word pairs in n given by,

$$N = \frac{n\ \ (n+1)}{2} \tag{3.5}$$

We use random walks through the rows to generate a constant number of poems for each template. Firstly, the total number of combinations possible for n words in a poem from a set of m potential replacements for each word is n^m. This value can be very high for longer poems and higher number of combinations also mean that larger number of poems need to be evaluated and processed in the steps that follow. Random walks also add a necessary element of randomness to the generated poems so that when the same template is used on the same theme different results are obtained.

3.4.2 Fixed Verse Constraint Evaluator

The poems generated in the previous steps are evaluated against the poetic constraints of the type of fixed verse chosen because there are possible violations during when words are randomly selected. We eliminate the poems which violate the constraints. We can choose to relax constraints like word stresses in interest of fewer poems to consider.

The constraints can be further relaxed for fixed verses like Haikus which are not expected to obey the traditional 5-7-5 pattern in the English language. Some impose an overall count of 17 syllables without stress to number of syllables in each line while others impose an upper bound of syllables to 17.

3.4.3 Poem Structure Evaluator

In addition to fixed verse structure evaluation, we also count the total number verse syntax violations. The generated poems are POS-tagged with our LingPipe HMM tagger. The POS-tagged sentences are checked for the number of mismatches in the POS-tags obtained from the template and latest POS-tags. This step removes

poems which are particularly hard to read due to multiple grammatical violations. If the number of mismatches occur for more than 40% of the words, we eliminate the poem.

The POS-tagging process has an accuracy of 96% for natural languages [55]. However, with poetic text, we observe the accuracy to be around 85% as a majority of poems violate grammatical structures for conciseness among other reasons. Hence, we do not impose a strict constraint on POS-tag violations. We discuss the problem of POS-tagging for poems in Chapter 6.

3.5 Poem Aggregation and Output

The steps of generation and evaluation are performed for each template poem chosen from the poetic structure repository. While we arrive at a "good" poem for each template, we use multiple templates to increase the chances of getting a good set of poems to choose from. The choice on the number of templates to be used for each poem request made by the user depends on the overall requests that can be handled by the implemented solution. For the purpose of this thesis, this value is set to 10.

At the end of the evaluation stage we either output the set of generated poems from each template or choose between poems from different templates by sorting the poems by their internal similarity score, S_I, and randomly choosing one of the poems near the middle. We do this because we find that poems with very high internal relatedness score tend to lie very close to the theme which makes them very artificial.

In the following chapter we set the hypotheses to evaluate the poetry generated by our method and describe a user study to test the hypothesis.

Chapter 4

USER STUDY

To determine *AutoPoe*'s success in generating poetry that is comparable to human composed poetry, we designed a comprehensive user study.

4.1 Hypotheses

We set two hypotheses to test using our user study. We set the first test in order to measure the existence of a bias in rating computer generated versus human composed poetry. The second hypothesis testing is done to test if computer generated poetry is indistinguishable from human composed poetry.

4.1.1 Measuring Bias

We wanted to test the presence of bias in rating poetry based on its origin. There would be a bias if the poem is rated based on the awareness of who composed the poem. For example, bias here refers to the behavior in Turing tests where a poem is rated just on the basis of the user's guess of it being a computer generated poem. The same would apply to high ratings for human poems. We employ an independence test to check if the ratings are consistent in the absence of knowledge about the origin of the poem. We assume that users who know the origin of the poem do not have a bias when rating the poem because we need a frame of reference for measuring the bias.

We believe that measuring bias is important for our project for two reasons. First, Turing tests check how indistinguishable the computer generated poems are from human composed poems. The presence of a bias would question the actual rating of the poem in the Turing test. Secondly, we intend to include the results from our

user study to further improve the system to generate better results. If the ratings are compromised by a bias then we cannot change the parameters used during generation to perform these improvements.

To test if the ratings for computer generated poems in a Turing test have a bias, we check for independence in poem rating both inside and outside the Turing tests. For this we set the following hypothesis H_A, where H_{0A} denotes the null hypothesis and H_{1A} denotes the alternate hypothesis.

H_{0A} : The ratings of computer generated poems in the Turing test and outside the Turing test have similar distributions.

H_{1A}: The ratings of computer generated poems in the Turing test and outside the Turing test have different distributions.

We will analyze ratings provided by users and perform the Pearson's χ^2 test for independence [77] to test the independence and reject/accept the hypothesis.

4.1.2 Turing Test

To test if the poems generated by *AutoPoe* are indistinguishable from human composed poetry we set a Turing test where we ask the users to guess if the presented poem had been written by a human or a computer. For this we set the following hypothesis H_B, where H_{0B} denotes the null hypothesis and H_{1B} denotes the alternate hypothesis.

H_{0B} : Computer generated poems are clearly distinguishable from Human composed poems.

H_{1B}: Computer generated poems are indistinguishable from Human composed poems.

We will accept/reject this hypothesis by analyzing the success of participants in identifying the source of the poems correctly.

4.2 User Study Design

We designed a user study to test the above hypotheses which involved participants to fill out an in-person questionnaire. We preferred to do the study in-person as opposed to hosting it online, to prevent people from searching the origin of the poem using the internet. Many human composed poems we used for our study could be found if searched.

We designed the questionnaire to be anonymous to obtain a larger number of participants for our study. We asked participants to provide their feedback on five to six questions which are used in evaluating poetry[38]. The questionnaires contained questions on both the form and content of the poems. For the Turing test we asked members to guess who was most likely to have composed the poem. All questions except the rating had a don't know option to choose in case there wasn't evidence against either.

The questionnaire conformed to the requirements of by the Institutional Review Board at Arizona State University. The participants were recruited by consent and all the tests were held in a university library or a class environment. No incentives were offered or provided for taking the user study. The IRB approval, consent form and questionnaire can be found in Appendix A, B and C.

4.2.1 Control and Experimental Group

To test hypothesis H_A we divide the questionnaire into two groups: a control group and an experimental group. The questionnaires for the control group involved evaluation of 3 human composed poems and 3 computer generated poems. Each poem was labeled HUMAN COMPOSED or COMPUTER GENERATED so that the participant in the control group knew the origin of the poem. We placed the human

composed poems before the computer generated poems so that the participant knew what scale the computer generated poems were being compared against.

The Turing test was conducted for the participants in the experimental group. The questionnaires for the experimental group contained 6 poems which was collection of a random number of computer generated poems and human composed poems. Neither the source/composer of the poem nor the number of computer generated poems was revealed to the participant in the experimental group.

4.2.2 Expert and Non-expert Group

We conducted the study for both experts and non-experts to obtain subjective and objective viewpoints. We collected demographic information from the users regarding their proficiency in English, academic background, interest in poetry and how many poems they read in a month.

We divided the user group into expert and non-expert groups based on their academic background and number of poems read per month. If the participant studied English literature at a university and if the number of poems read were greater than 15, the participant was considered an expert.

The study was held separately for expert and non-expert groups. Within each group the study was further divided into a control group and experimental group to account for bias in the study (if any) and to establish a baseline for responses.

4.3 Poem Selection

For the study we evaluated a total of 24 poems which included 12 human composed poems and 12 computer generated poems. Among the 12 poems in each category, 8 were haiku, 2 were limericks and 2 were four line verses in iambic pentameter. We choose haiku, limericks and iambic pentameter verse because they demonstrate fixed

verse generation with different constraints. Haikus have constraints on the number of syllables in the entire poem while Limericks demand a rhyming scheme A-A-B-B-A. Iambic pentameter verses have constraints on stress patterns.

Each participant was provided with 6 poems in total, which contained 4 haikus, 1 limerick and 1 iambic pentameter verse, in the specified order. We favor haikus over longer poems as they take much less time to read and lessen the anticipated fatigue and the overall time taken to complete the study. All poems chosen for the study are included in Appendix D.

4.3.1 Human Composed Poems

We divided the source of human composed poems into two categories, professionals and amateurs. For professional poets, we find haikus from published books [86, 50, 97, 95], and rated poems from online sources. For amateur poetry, we randomly select four haikus from twitter data. We pick the two limericks at random due to unavailability of ratings.

4.3.2 Computer Generated Poems

For generating poems we set the α parameter at 0.75 and generate haikus, limericks and iambic verses for themes chosen from "love", "beauty", "winter", "summer", "life", "milk", and "blue". For computer generated poems, we split the 12 poems such that half of them are chosen at random and the remaining are chosen by a human based on their form and content to add a human element of selection. The poems generated by the system are sorted by their internal similarity (S_I) scores. We generate a total of 120 haikus of which a human selects four from them. After eliminating the top 25% and bottom 25% of the list, four haikus are chosen by the computer randomly among the remaining poems. Similarly we generate 20 limericks

		Number of Poems		No of Human-Computer poems			Sequence of poems in each set					
	Set ID	Human Composed	Computer Generated	Haiku	Limerick	Iambic	Poem #1	Poem #2	Poem #3	Poem #4	Poem #5	Poem #6
Control Group	A	3	3	2-2	1-0	0-1	hh1	hh5	hl1	mh1	mh5	mi2
	B	3	3	2-2	0-1	1-0	hh2	hh6	hi2	mh2	mh6	ml1
Experimental Group	C	6	0	4-0	1-0	1-0	hh3	hh4	hh7	hh8	hl2	hi1
	D	4	2	3-1	1-0	0-1	hh1	hh5	hh2	mh3	hl1	mi1
	E	3	3	2-2	0-1	1-0	hh6	hh3	mh7	mh4	ml2	hi2
	F	3	3	2-2	1-0	0-1	hh7	hh4	mh8	mh1	hl2	mi2
	G	2	4	1-3	0-1	1-0	hh8	mh5	mh2	mh6	ml1	hi1
	H	0	6	0-4	0-1	0-1	mh3	mh7	mh4	mh8	ml2	mi1

Figure 4.1: User Study Plan

and 20 iambic verses where one is selected by the human and one is selected by the computer.

4.4 Study Plan

25% of the study population forms the control group to determine the bias while the remaining form the experimental group to take the Turing test. We create eight sets of questionnaires of the form given in Figure 4.1. Sets A and B form the control group and rest of them form the experimental group. For each set that is answered in the user study, there is a total coverage of two reviews per poem. The poem id is a 3 character id where first character indicates the composer, 'h' for human and 'm' for machine/computer and second character type of poem indicates type of poem, i.e., 'h', 'l' and 'i' for haiku, limerick and iambic pentameter verses. The third character denotes the sequence number of the poem. We specifically design some sets to contain varying numbers of human composed and computer generated poems. For instance, set C contains only human generated poems and set H contains only computer generated poems.

In the next chapter we discuss the hypothesis testing and results of the user study.

Chapter 5

RESULTS AND DISCUSSION

For reference, all poems used for the study have been included in Appendix D. Forty subjects participated in the user study which included 12 experts who had an academic background in English literature. 28 participants belonged to the non-expert group. 11 participants from the expert group answered the questionnaire in a classroom environment and 1 answered it in a library environment. Most participants from the non-expert group took the study in the library environment. Although the study was not a timed test, we observed that the time taken to answer the questionnaire took approximately 10 minutes on average.

5.1 Measuring Bias : H_A

First we analyze the results to test the hypothesis H_A. We measure the bias in the study by calculating the variance in ratings measured between the control and experimental group. If there is a bias in the ratings, we would find that the ratings from both the groups for individual poems are independent at a high significance level. The Pearson's χ^2 test for independence for the expert group and non-expert group is included in Figures 5.1 and 5.2 respectively.

The Pearson's χ^2 value is calculated using Equation 5.1.

$$\chi^2 = \sum_{i=1}^{n} \frac{(O_i - E_i)^2}{E_i} \tag{5.1}$$

where,

χ^2 is the Pearson's cumulative test statistic,

Expert Group		Individual Ratings				Rating Averages			Chi-Sq
Method	Code Name	#1	#2	#3	#4	Overall	Control *E*	Expmntl *O*	*(O-E)sq/E*
Human	hh1	5	1	2	4	3.0	3.5	2.5	0.28571
	hh2	4	3		5	4.0	4	4	0.00000
	hh5	5	2	3	5	3.8	4	3.5	0.06250
	hh6	3	1		1	1.7	3	1	1.33333
	hl1	5	3	4	5	4.3	4.5	4	0.05556
	hi2	3	4		2	3.0	3	3	0.00000
Computer	mh1	5	3	2		3.3	3.5	3	0.07143
	mh2	4	4			4.0	4	4	0.00000
	mh5	5	2	2		3.0	3.5	2	0.64286
	mh6	3	3			3.0	3	3	0.00000
	ml1	4	4			4.0	4	4	0.00000
	mi2	5	3	3		3.7	4	3	0.25000
							Chi-squared value =		2.70139

Set A Set B

Figure 5.1: χ^2 Test for Independence to Detect Bias in the Expert Group

Non-expert Group		Individual Ratings								Rating Averages			Chi-Sq
Method	Code Name	#1	#2	#3	#4	#5	#6	#7	#8	Overall	Control *E*	Expmntl *O*	*(O-E)sq/E*
Human	hh1	3	2	1	4	4	2			2.67	2.67	2.67	0.00000
	hh2	3	3	2	3	3	3	3		2.86	2.75	3.00	0.02273
	hh5	4	4	3	4	4	4			3.83	3.67	4.00	0.03030
	hh6		4	3	5	4	1	1		3.00	2.67	3.33	0.16667
	hl1	4	2	3	3	2	3			2.83	3.00	2.67	0.03704
	hi2	4	5	4	5	4	3	3		4.00	3.75	4.33	0.09074
Computer	mh1	3	2	1	3	1	2	1	1	1.75	1.50	2.00	0.16667
	mh2	3	1	1	3	2	3	3		2.29	2.25	2.33	0.00309
	mh5	2	2	2	1	3	4			2.33	2.33	2.33	0.00000
	mh6	4	4	3	2	4	3	2	3	3.13	3.25	3.00	0.01923
	ml1	1	3	3	5	3	2	3	1	2.63	2.50	2.75	0.02500
	mi2	3	4	4	4	4	2	3		3.43	3.50	3.33	0.00794
											Chi-squared value =		0.56940

Set A Set B

Figure 5.2: χ^2 Test for Independence to Detect Bias in the Non-expert Group

O_i is the observation value, i.e., the experimental group rating for poem i,

E_i is the expected value, i.e., the control group rating for poem i, and

n is the number of observations, here number of poems.

The number of degrees of freedom df for the distribution is given by Equation 5.2.

$$df = (N_c - 1)(N_r - 1) = (2 - 1)(12 - 1) = 11 \tag{5.2}$$

where,

N_c is the number of columns in Figure 5.1 and 5.2, and

N_r is the number of observations in Figure 5.1 and 5.2.

The number of columns is 2 because we consider rating averages and the number of rows is the number of poems, i.e., 12. We find that for 11 degrees of freedom, the upper-tail critical values give us a critical value of 3.053 at 1% significance level. Our χ^2 values for expert (2.70139) and non-expert group (0.56940) are less than the p-value at 1% significance level (3.053). Hence we fail to reject the null hypothesis H_{0A}. Our results suggest that user's ratings are not biased by the user's knowledge of the authorship of the poem (computer or human). Given the limited size of the study, more work would be needed to have more confidence in the absence of such bias.

5.2 Turing Test Results

To test the hypothesis H_B, we will look at the accuracy with which each user guessed the poem's author as being human or computer.

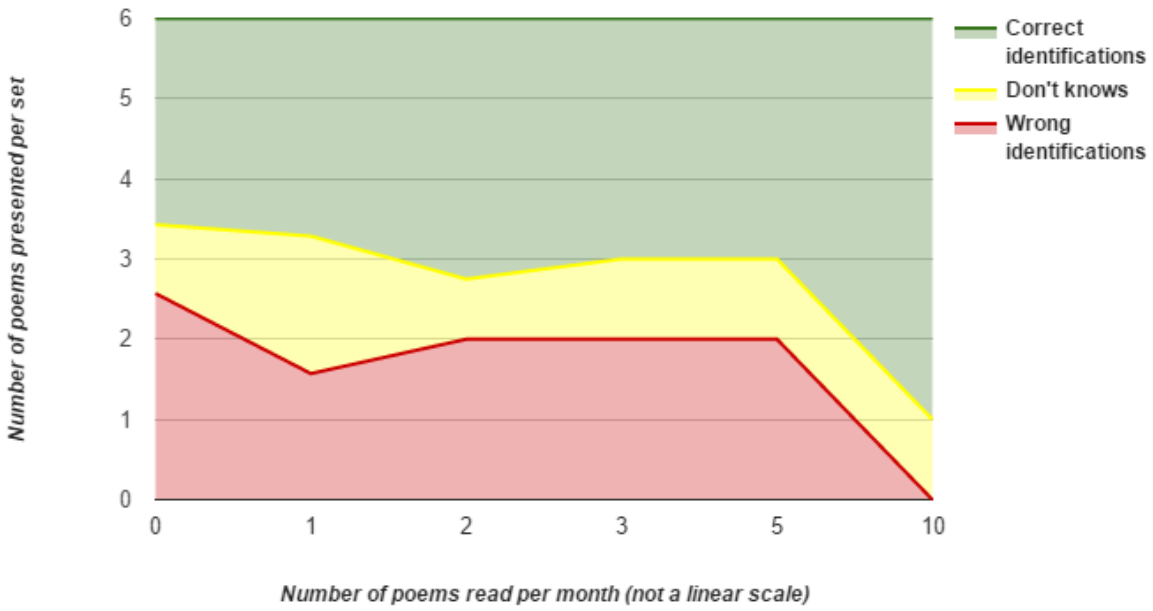

Figure 5.3: Non-expert Group's Accuracy for the Turing Test

5.2.1 Non-Expert Group

Twelve non-experts answered that they read 0 poems per month. The number of poems read by non-experts per month averaged 1.28. One participant from the non-expert group reported that he/she read 10 poems per month which was recorded to be highest.

Figure 5.3 shows the accuracy of the non-expert group arranged by the number of poems read per month. We clearly notice that in the non-expert group, frequent readers of poetry were more likely to predict the author of the poems correctly.

Figure 5.4 shows the breakdown of the errors in identification among the non-expert group. A majority of the errors belonged to the case where a computer generated poetry was incorrectly classified as human composed. None of the participants were able to correctly select all poems' sources, although one user correctly selected five poems and answered one as *"Don't know"*.

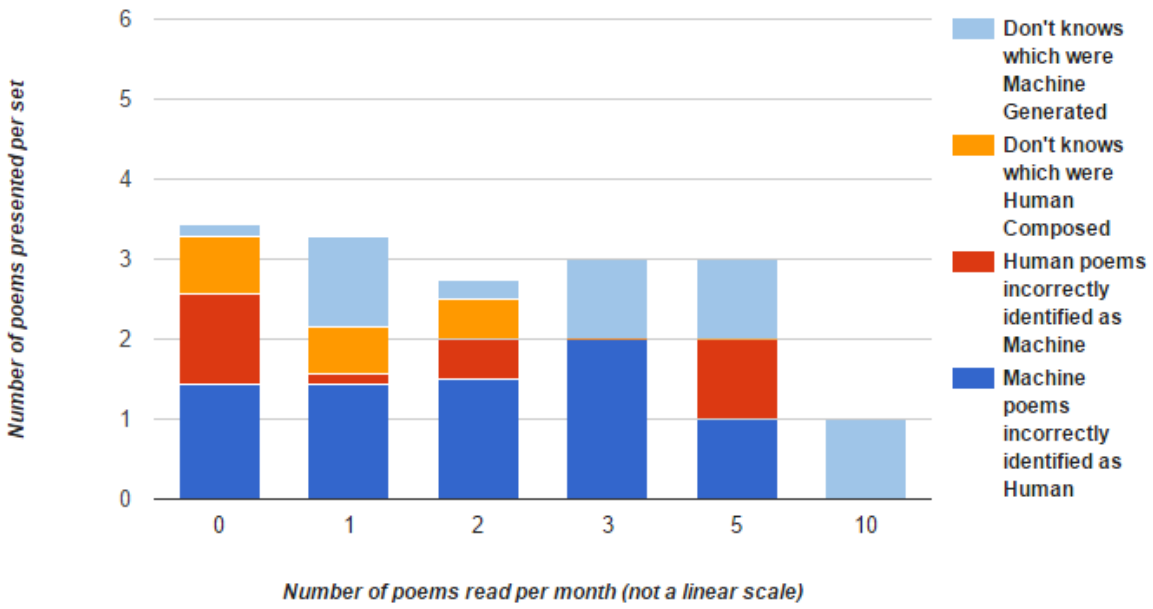

Figure 5.4: Identification Failures by the Non-expert Group

In all, of the 126 attempts at the question, non-experts incorrectly identified 29 poems as being human although they were computer generated and 12 human poems were incorrectly identified as computer generated for a total of 41 wrong identifications (32%). The overall correct identifications were only 61 (48%).

Of the 67 total computer poems in the non-expert's questionnaires, 29 were incorrectly identified as human (43%) and 13 were answered with a *"Don't know"* (19%). So, non-experts were successful about 38% of the time when it came to correctly identifying if a poem was computer generated. Among the 29 incorrectly identified poems, 14 were randomly chosen while 15 were chosen by a human. Hence, selection by the human only had a marginal advantage. To our surprise, while we expected the larger word count in limericks and iambic pentameter verses to significantly reduce the chances of it being passed as human, almost 50% of the 29 incorrectly identified poems included longer poems although they formed only 33% of the total number presented for evaluation.

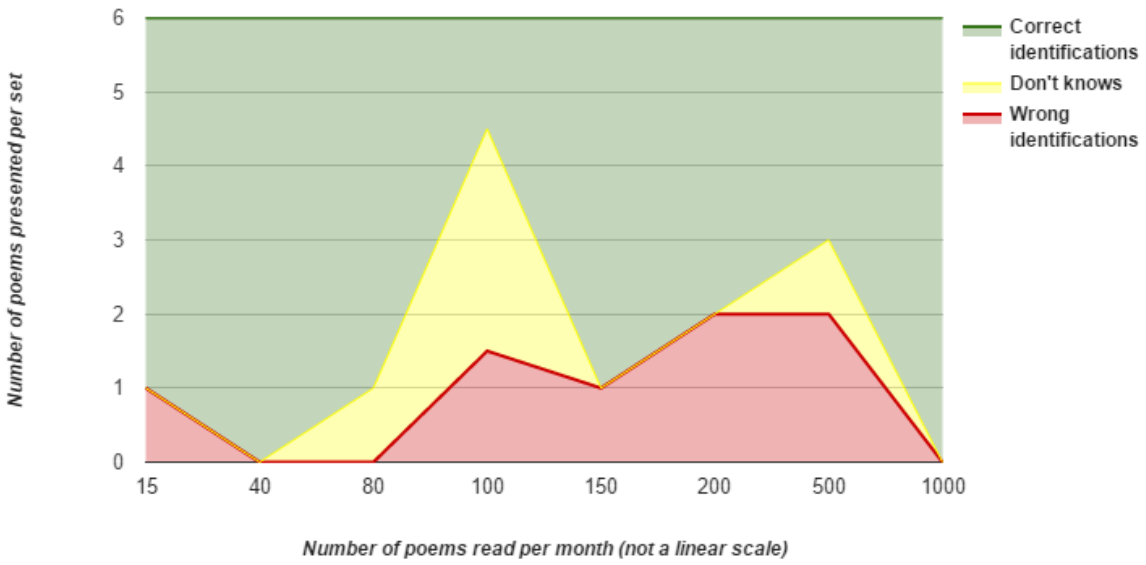

Figure 5.5: Expert group's accuracy for the Turing test

5.2.2 Expert Group

There was a clear difference in the number of poems read per month on an average between experts and non-experts. The lowest number of poems read by experts was 15 while the highest was 1000. The average was 195 poems per month. Figure 5.5 shows the expert group's accuracy in identifying the source of poem accurately. Among experts, the number of poems read per month had little effect on the accuracy. Although the participant who had read the most number of poems selected all poems' source accurately, another participant who reported as reading 40 poems per month was similarly successful.

Among the 54 poems reviewed by experts, only 9 were identified incorrectly(16%) and 8 were answered with *"Don't know"* (19%). So, experts were accurate in their identification 65% of the time. Of the 23 computer generated poems that was presented to experts, only 3 were incorrectly identified as human (13%) and 5 were reported as *"Don't know"* (21%). Surprisingly, one published haiku written by a

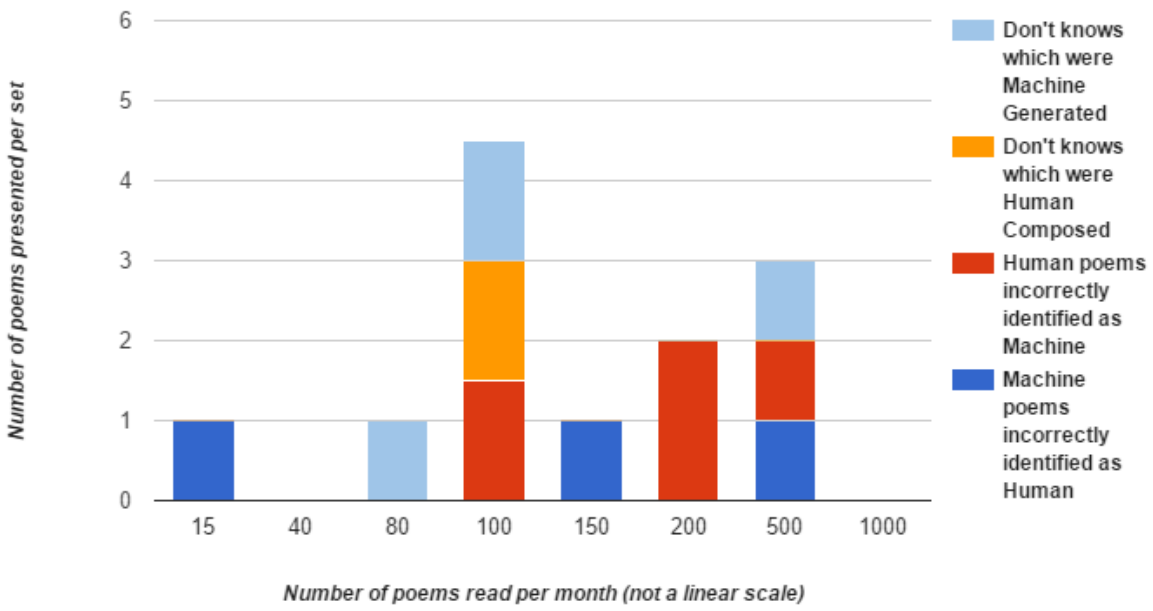

Figure 5.6: Identification failures by the expert group

professional was incorrectly identified as computer generated by two experts. Three among the 4 amateur haikus was incorrectly identified as computer generated. Figure 5.6 shows identification errors among the expert group.

5.3 Expert Group Feedback

In addition to identifying whether the poems were written by computers or humans, we also received feedback from the participants on the form and content of the poetry. For the purpose of conciseness and in-depth analysis, we only discuss the results from our expert group feedback. The summary of feedback from experts about presence of poetic devices, imagery, clichés and artificiality in the human composed poems is included in Figure 5.7. The rows represent the poems used in the study while the columns represent the feedback collected for each poems. The questions used to collect the feedback are available for reference in Appendix C. The ratings use a scale of 1 to 5, while the other four responses use binary scale of 0 and 1. Ideally,

Human Composed	Poem ID	Avg. scores for user feedback for the poems				
		Ratings	Poetic Devices	Emotion/ Imagery	Cliches	Artificial
Published Poetry	hh1	3.00	0.50	1.00	0.25	0.25
	hh2	4.00	1.00	1.00	0.00	0.00
	hh3	2.67	1.00	1.00	1.00	0.00
	hh4	3.50	1.00	1.00	0.50	0.00
	hl1	4.25	1.00	1.00	0.50	0.00
	hi1	3.00	1.00	1.00	1.00	0.50
Amateur Poetry	hh5	3.75	1.00	1.00	0.00	0.00
	hh6	1.67	0.67	0.67	0.33	0.67
	hh7	2.50	1.00	0.50	1.00	0.50
	hh8	3.00	1.00	1.00	0.00	0.50
	hl2	3.50	1.00	1.00	0.00	0.00
	hi2	3.00	1.00	0.67	0.67	0.33
	All	3.15	0.93	0.90	0.44	0.23
	Published	3.40	0.92	1.00	0.54	0.13
	Amateur	2.90	0.94	0.81	0.33	0.33
	Haikus	3.01	0.90	0.90	0.39	0.24
	Limericks	3.88	1.00	1.00	0.25	0.00
	Iambic	3.00	1.00	0.83	0.83	0.42

Figure 5.7: User Reported Ratings and Answers for Human Composed Poetry

a good poem would be rated 1 for poetic devices and emotion/imagery, and 0 for clichés and artificiality.

While professional poetry scored 17% higher in ratings over amateur poetry, it was rated as being more clichéd in one of the haikus and verses. Limericks scored high in all areas and the haiku category was second. Haikus were rated better in the four areas for professional published poetry than for amateur poetry.

The feedback from experts about presence of poetic devices, imagery, clichés and artificiality in computer generated poems is presented in Figure 5.8. Although we expected to find that human selected poems tend to do score better in ratings over randomly selected poems, we find that randomly selected poems by the computer do much better than human selected poems. It is also surprising to find that randomly

Computer Generated	Poem ID	Avg. scores for user feedback for the poems				
		Ratings	Poetic Devices	Emotion/ Imagery	Cliches	Artificial
Human Selected	mh1	3.33	0.67	1.00	0.00	0.00
	mh2	4.00	1.00	1.00	0.00	0.00
	mh3	1.33	0.33	0.33	0.67	0.67
	mh4	1.00	1.00	0.00	1.00	1.00
	ml1	2.50	0.50	0.50	0.00	0.50
	mi1	1.00	0.67	0.33	0.67	0.67
Random Selection	mh5	3.00	1.00	0.50	0.00	0.67
	mh6	3.00	1.00	0.50	0.50	0.50
	mh7	1.67	0.67	0.33	1.00	0.67
	mh8	3.50	1.00	1.00	0.00	0.50
	ml2	2.33	0.67	0.33	0.33	1.00
	mi2	3.67	1.00	1.00	0.00	0.50
	All	2.53	0.79	0.57	0.35	0.56
	Selected	2.19	0.69	0.53	0.39	0.47
	Random	2.86	0.89	0.61	0.31	0.64
	Haikus	2.60	0.83	0.58	0.40	0.50
	Limericks	2.42	0.58	0.42	0.17	0.75
	Iambic	2.33	0.83	0.67	0.33	0.58

Figure 5.8: User Reported Ratings and Answers for Computer Generated Poetry

selected poetry scored better in 4 of the 5 areas. Among computer generated poetry, haikus scored better in ratings over limericks and iambic pentameter verses while it was reported to contain more clichés than others.

5.4 Turing Test : H_B

To accept/reject our second hypothesis, i.e., to determine if computer generated poems are indistinguishable from human generated poems or not, we will analyze our Turing test results and expert feedback.

5.4.1 *Identification Errors*

Comparing the success in identifying the computer generated poems, we find that non-experts had very low success of about 38% and were unsuccessful with 46% of the poems. In case of non-experts we can conclude that computer generated poems were largely indistinguishable.

In contrast, participants in the expert group correctly identified the computer generated poems 65% of the time. Additionally, failures in identification were only in 16% of the cases. Hence, we can conclude that in case of experts, a majority of the computer generated poems were distinguishable.

5.4.2 *Rating Comparison*

Comparing the ratings for human composed poems and computer generated poems from Figure 5.7 and Figure 5.8, we find that computer generated poems received lower scores in terms of artistic ratings. Overall, human composed poems were rated almost 30% higher than computer generated poems.

Based on our comparisons of identification errors among experts and non-experts and ratings from experts we fail to reject the null hypothesis H_{0B} and accept that the computer generated poetry is distinguishable from human generated poetry. To reject the null hypothesis such as this, it is imperative that the generation method be indistinguishable in both expert and non-expert groups. We would expect to see the percentage of incorrect identifications in both groups to exceed the 50% mark.

5.5 Summary

Failing to reject the null hypothesis H_{0B} is not entirely discouraging as we had set an ambitious goal of passing the Turing test with the hypothesis. In AI and

NLP research we find this be a very ambitious goal. A 46% and 19% failure in identifying computer generated poetry among non-experts and experts respectively is an improvement over some of the earlier methods which used Turing tests.

For instance, the *Gaiku* system [71] used a Turing test to identify computer generated haiku among a non-expert group. The participants incorrectly identified the sources of haiku 34% of the time. In comparison, non-experts in our user study failed to identify the sources of haiku 53% of the time and the failure in identifying the sources of the poems irrespective of the type, was 52%. Hence, *AutoPoe* has an improvement of 19% compared to the *Gaiku* system.

The POS-tag based poetry generation system [1] in Basque language used a Turing test for their poems on two linguists. This system uses three different methods for word substitutions. We compare our user study only with the first method because in the second and third methods, the words replaced from the original poem are restricted to two POS-tags. In *AutoPoe*, the words in the original template are replaced irrespective of the POS-tags. In their user study, 18% of the computer generated poems passed as human composed. In comparison, only 13% of the computer generated poems pass as human composed in our system. But an additional 22% of the computer generated poems in our method failed to be identified as either human composed or computer generated which makes our system better by 17%.

In the next chapter we will discuss some of the limitations observed in our system and suggest methods to overcome those limitations. We also list enhancements for the existing system and some interesting challenges that needs to be addressed in the future.

Chapter 6

LIMITATIONS AND FUTURE WORK

In this chapter, we discuss some of the shortcomings of our system and suggest possible solutions. In the later sections we discuss enhancements and interesting challenges in NLP and NLG that could be addressed in the future.

One of the foremost limitations is in the user study where we had only 40 subjects participate. The statistical significance tests, such as the Pearson's χ^2 test are believed to be less reliable in such low sample sizes. However, we overcome this shortcoming by taking the p-value at 1% significance level.

6.1 Error Analysis

A majority of the errors introduced in the final poems generated were due to erroneous results from the database. For example, single letter words such as "s" or "r" which are not words enter the dictionary. Dictionary verification of new words is a difficult task as the dictionary we use is a static 135,000 word dictionary. Yet new words that are used frequently are introduced every day as part of colloquial text have to be added to the dictionary dynamically.

Most errors introduced in our dictionary are due to parsing and POS-tagging errors. These errors are introduced due to absence of punctuation and resulting ambiguity in sentence tokenization. Fortunately, around half the errors seemed to go unnoticed in the final poems, as grammar is relaxed in poetry. However, POS-tagging which result in three or more adjectives in a row are more often regarded as erroneous rather than artistic signatures.

Currently new words are added to the dictionary after it appears over 5 times

with a single part of speech. Best effort rhyme, syllable count and stress calculations are made before it is indexed to the *AutoPoe* dictionary. This method adds words which are not words in the dictionary but are an output of inaccurate tokenizations.

The CMU pronouncing dictionary is a phonetic dictionary and not a phonemic dictionary. Hence, it does not take into account various pronunciations in native accents and foreign accents. We notice errors in one of limericks selected for the user study where "violet" rhymes with "granite" and "desperate". Since it is a human compiled dictionary by volunteers across the world, it introduces a few errors.

6.2 Computational Complexity and Optimization

Our system has been designed as a proof of concept for using neural word embeddings as a method to generate poetry. Much of its focus has been generating poetry that is indistinguishable from human composed poetry with less focus on scalable efficiency at this stage. Our method for generating a poem for a given template and a given theme has a computational complexity in the lower order polynomial in n, i.e., $O(n^2)$ where n is the number of words in the template poem. The verse generation step has a complexity of $O(n)$ where candidate words for each word position is compared with the desired theme to calculate the replacement R_w scores. The evaluation step has a complexity of $O(n^2)$ where internal similarity (S_I) scores are calculated.

However, with increase in number of theme words where each candidate word needs to be checked for its similarity with each theme, the system would need much longer to generate a poem. The operational efficiency of our poetry generator could be improved by making some optimizations in the random walks. When using multiple themes, a centroid distance measure could be used to aggregate the distances between the theme words.

6.3 NLP Challenges in Poetry

"Poetry is language at its most distilled and most powerful."

Rita Dove

Current models for language processing and sentiment analysis find figurative language hard to identify, let alone generate. Poetry is rich in metaphors and figurative meanings when compared to natural conversational language. Hence, poetry offers an interesting set of challenges in natural language processing research.

6.3.1 POS-tagging Poetry

Poems often have a brief and condensed form of sentence. The condensed version of verse often break rules of grammar for effect. Many term this privilege as a poetic license. Poets sometimes omit punctuation for effect or as part of their artistic style which does have the intended effect when being read but introduces challenges in sentence parsing. Each sentence may be spread over multiple lines of a verse. This poses a challenge in POS-tagging such sentences as tokenizing sentences over multiple lines in the absence of punctuation is a difficult task.

We address this problem to some extent in our system by using a unidirectional POS-tagger and treating every line as an individual sentence. The POS-tagging results were analyzed and found to be more successful than bi-directional taggers. Assigning more weights to observational probabilities/emissions than sequential probabilities for predicting POS-tags obtained mixed results. Hence, a dedicated effort to address auto-punctuation and sentence tokenization would be interesting to pursue.

6.3.2 Verse to Prose and Word Sense Inclusion

Sentence expansion of condensed verses to aid sentiment analysis would be an invaluable research contribution. Many corpus based methods we have studied for poem generation start with prose and condense them into poetic verse. An effort to lossless reversal of the process while maintaining the original meaning would be an interesting research experiment.

AutoPoe uses words and their POS-tags for setting the poem template, and also during the stochastic search process. Errors in the method would greatly reduce if we were to store the multiple senses in which a word is used within a particular part of speech. However, it is a difficult problem to solve as there is no corpora in which word senses have been annotated. During initial experiments, including the semantic hierarchy of words during search apparently has a negative impact on generation. Finding similar words by hierarchies led to predictable results which were similar to the original poems. We suspect that incorporating word sense may have a similar negative effect on the quality of poems.

6.3.3 Serendipity and Imagery

Recent research in computational creativity have acknowledged that in a creative system, serendipity plays an important role [78]. Randomness and unintentional detours are instrumental in introducing serendipity.

In *AutoPoe*, we introduce these elements in two places. We introduce randomness during our random walks through the multiple combinations of candidate words. A detour is observed in most cases where the theme provided by the user is unrelated to the poem template chosen and the resulting poem is closer to the theme than the original poem depending on the chosen value of α. Introducing an additional

element of detour within the confines of the poem by randomly choosing one of the word positions to mutate towards a randomly selected theme by the computer may improve serendipity.

Four users from our study commented that some of the computer generated poems (which they identified correctly) lacked imagery. Imagery refers to semantics in text which trigger mental images or generally one of the five senses. We believe that introduction of imagery in computational creativity is a problem similar to serendipity and requires a larger focus on building pragmatic knowledge. NLP has for a long time listed its major challenges in the following order of complexity: lexical (syntactic) analysis, semantic analysis and pragmatic analysis. Future work should focus on building the social contextual information and incorporate them into the generation process.

6.4 Experimental Enhancements

The proposed system has errors that needs to be addressed to some extent before enhancements are introduced. However, we list a number of enhancements that are interesting NLP challenges.

6.4.1 True Haikus and other Forms of Poetry

If we were to adopt the traditional definition of haiku, the generated poems should deal with "Nature and human nature" and senryus deal with human processes. Hence in addition to the user provided theme, we should also find similarity to human traits and nature related attributes.

This thesis focused on generating poems belonging to a few fixed verse types. Extending the types of fixed verse to other forms such as tanka, ballad, sonnet etc. would be interesting. Neural word embedding methods to generating free verse would

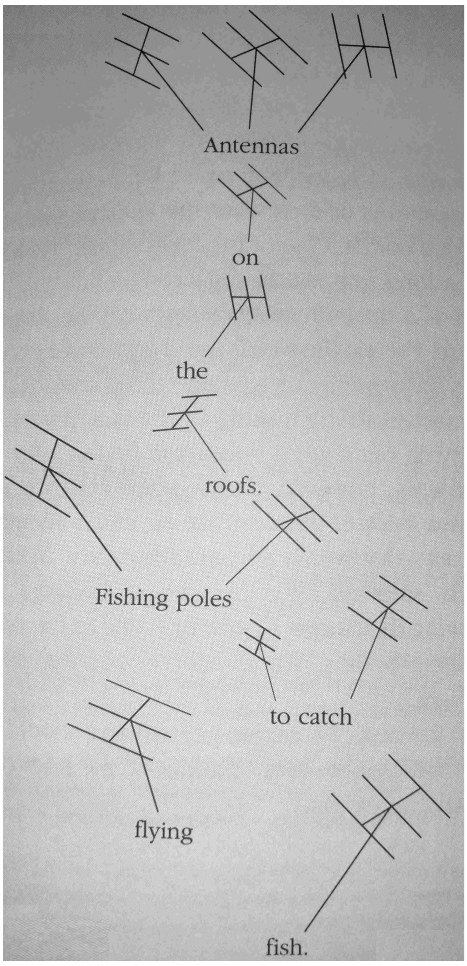

Figure 6.1: An Example of Concrete Poetry, "Catchers" by Robert Froman [33]

require a different approach where we take an iterative generation cycle for generating poetry with a lower value of α for slow mutations.

6.4.2 Concrete Poetry

An experimental enhancement to our poetry generation system would be to generate concrete poetry [28] which have both language and visual cues. A simple example for concrete poetry has been shown in Figure 6.1.

6.4.3 Feedback-driven Generation

Extending the generation process to be driven by user ratings would be a welcome addition to this research. This includes encoding some of the feedback given by the expert group by setting the value of α to a lower value and using an iterative method to arrive at the final version of the poem. Deploying the application in social media is a viable option to receive input from the user and generate poetry. Although, there would only be one parameter in the form of "likes/dislikes" to determine if the generation was "good", we believe that the overall model can be integrated with social media.

Chapter 7

CONCLUSION

In this thesis we present a novel approach to poetry generation using neural word embeddings. During the course of generation, we developed a two layered poetry classifier which was observed to have an accuracy of 92%. The first layer uses a lexicon based method to classify poems into various types based on form and structure and the second layer uses a supervised classifier to classify poems into subtypes based on content. Using the fixed verse poetry generator, we demonstrated its capabilities in generating text within the constraints imposed by the type of poetry such as rhyme, rhythm, syllable counts and stress patterns. We generate haikus, limericks and iambic pentameter verses and we address various NLP and NLG challenges faced in the process of generation. We test the generated poems by designing a Turing test based user study which includes both experts and non-experts.

From our user study we find that, only 38% computer generated poems were correctly identified by non-experts. Hence, we conclude that computer generated poems were largely indistinguishable from human composed poems among the non-experts. Among experts, 65% of the computer generated poems were correctly identified. Although the system does not pass the Turing test conclusively, the results from the Turing test were found to have an improvement of 17% over previous methods which use similar Turing tests to evaluate poetry. Some additional research on introducing serendipity, metaphorical and figurative language into our poetry we can generate diverse set of poems. With a conservative approach to error handling in the dictionary, we hope to achieve better results in generating poetry which is closer to being indistinguishable from human composed poetry among experts too.

REFERENCES

[1] M. Agirrezabal, B. Arrieta, A. Astigarraga, and M. Hulden, "POS-tag based poetry generation with WordNet," in *Proceedings of the 14th European Workshop on Natural Language Generation*, 2013, pp. 162–166.

[2] R. Agrawal, R. Srikant *et al.*, "Fast algorithms for mining association rules," in *Proceedings of 20th International Conference of Very Large Data Bases, VLDB*, vol. 1215, 1994, pp. 487–499.

[3] Alias-i, "Lingpipe 4.1.0," http://alias-i.com/lingpipe, 2008.

[4] Apache, "Apache Lucene," https://lucene.apache.org/, 2010.

[5] E. Asgari and M. R. Mofrad, "Continuous distributed representation of biological sequences for deep proteomics and genomics," *PloS one*, vol. 10, no. 11, p. e0141287, 2015.

[6] R. W. Bailey, "Computer-assisted poetry: The writing machine is for everybody," *Computers in the Humanities*, pp. 283–295, 1974.

[7] Y. Bengio, H. Schwenk, J.-S. Senécal, F. Morin, and J.-L. Gauvain, "Neural probabilistic language models," in *Innovations in Machine Learning*. Springer, 2006, pp. 137–186.

[8] R. Bhatnagar, "Pentametron," *http://pentametron.com/*, 2012.

[9] L. Bloom, *Language development*. University of Chicago Press Chicago, 1975.

[10] C. Browne, *Evolutionary game design*. Springer Science & Business Media, 2011.

[11] M. Campbell, A. J. Hoane, and F. H. Hsu, "Deep Blue," *Artificial intelligence*, vol. 134, no. 1, pp. 57–83, 2002.

[12] G. Carenini and J. C. K. Cheung, "Extractive vs. nlg-based abstractive summarization of evaluative text: The effect of corpus controversiality," in *Proceedings of the Fifth International Natural Language Generation Conference*. Association for Computational Linguistics, 2008, pp. 33–41.

[13] J. Carpenter, "Electronic text composition project," *The Slought Foundation*, 2004.

[14] W. Chamberlain and J. Hall, *The Policeman's Beard is Half Constructed: Computer Prose and Poetry by Racter;the First Book Ever Written by a Computer; a Bizarre and Fantastic Journey Into the Mind of a Machine*. Warner Books, 1984.

[15] N. Chomsky, *Aspects of the Theory of Syntax*. MIT press, 2014, vol. 11.

[16] S. Colton, "Refactorable numbers-a machine invention," *Journal of Integer Sequences*, vol. 2, no. 99.1, p. 2, 1999.

[17] S. Colton and B. P. Ferrer, "No photos harmed/growing paths from seed: an exhibition," in *Proceedings of the Symposium on Non-Photorealistic Animation and Rendering*. Eurographics Association, 2012, pp. 1–10.

[18] S. Colton, J. Goodwin, and T. Veale, "Full FACE poetry generation," in *Proceedings of the Third International Conference on Computational Creativity*, 2012, pp. 95–102.

[19] S. Colton and G. Sutcliffe, "Automatic generation of benchmark problems for automated theorem proving systems." in *International Symposium on Artificial Intelligence and Mathematics*, 2002.

[20] S. Colton and G. A. Wiggins, "Computational creativity: The final frontier?" in *European Conference on Artificial Intelligence*, vol. 12, 2012, pp. 21–26.

[21] S. Colton, *Automated theory formation in pure mathematics*. Springer Science & Business Media, 2012.

[22] J. Corneli, A. Jordanous, R. Shepperd, M. T. Llano, J. Misztal, S. Colton, and C. Guckelsberger, "Computational poetry workshop: Making sense of work in progress," in *Proceedings of the Sixth International Conference on Computational Creativity June*, 2015, p. 268.

[23] J. Corneli, A. Pease, S. Colton, A. Jordanous, and C. Guckelsberger, "Modelling serendipity in a computational context," *arXiv preprint arXiv:1411.0440*, 2014.

[24] P. Cramer, *Word association*. Academic Press New York, 1968.

[25] Deeplearning4j, "Deeplearning4j: Open-source distributed deep learning for the JVM," http://deeplearning4j.org, 2015.

[26] Deeplearning4j, "word2vec skip-gram model," http://deeplearning4j.org/word2vec, 2015.

[27] B. Díaz-Agudo, P. Gervás, and P. A. González-Calero, "Poetry generation in COLIBRI," in *Advances in Case-Based Reasoning*. Springer, 2002, pp. 73–87.

[28] R. Draper, "Concrete poetry," *New Literary History*, vol. 2, no. 2, pp. 329–340, 1971.

[29] K. Ebcioğlu, "An expert system for harmonizing chorales in the style of JS Bach," *The Journal of Logic Programming*, vol. 8, no. 1-2, pp. 145–185, 1990.

[30] A. Esuli and F. Sebastiani, "SentiWordNet: A publicly available lexical resource for opinion mining," in *Proceedings of Language Resources and Evaluation Conference*, vol. 6. Citeseer, 2006, pp. 417–422.

[31] D. A. Ferrucci, "Introduction to "this is watson"," *IBM Journal of Research and Development*, vol. 56, no. 3.4, pp. 1–1, 2012.

[32] W. N. Francis and H. Kucera, "Brown corpus manual," *Brown University*, 1979.

[33] R. Froman, *Street Poems*. McCall Publishing Company, 1971.

[34] P. Gervás, "WASP: Evaluation of different strategies for the automatic generation of spanish verse," in *Proceedings of the AISB-00 Symposium on Creative & Cultural Aspects of AI*, 2000, pp. 93–100.

[35] P. Gervás, D. Sistemas, and I. Programacion, "Automatic generation of poetry using a CBR approach," *CAEPIA-TTIA 01 Actas Volumen I*, 2001.

[36] P. Gervás, "Generating poetry from a prose text: Creativity versus faithfulness," in *Proceedings of the AISB01: Symposium on Artificial Intelligence and Creativity in Arts and Science*, 2001, pp. 93–99.

[37] P. Gervás, "Modeling literary style for semi-automatic generation of poetry," in *User Modeling 2001*. Springer, 2001, pp. 231–233.

[38] U. T. Gibson, "On judging poetry : A level-headed means to evaluate poetry," http://www.chaparralpoets.org/judging.html, 2008.

[39] Gigaword Corpus, https://catalog.ldc.upenn.edu/LDC2003T05, 2016, accessed: 2016-03-21.

[40] Y. Goldberg and O. Levy, "word2vec explained: Deriving Mikolov et al.'s negative-sampling word-embedding method," *arXiv preprint arXiv:1402.3722*, 2014.

[41] P. G. Gómez-Navarro, "Un modelo computacional para la generación automática de poesía formal en castellano," *Procesamiento del lenguaje natural*, no. 26, pp. 19–26, 2000.

[42] U. Goswami, "Phonological skills and learning to read," *Annals of the New York Academy of Sciences*, vol. 682, no. 1, pp. 296–311, 1993.

[43] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten, "The WEKA data mining software: an update," *ACM SIGKDD explorations newsletter*, vol. 11, no. 1, pp. 10–18, 2009.

[44] J. Harris, " The Times Haiku," *http://haiku.nytimes.com/*, 2013.

[45] Z. S. Harris, "Distributional structure," *Word*, vol. 10, no. 2-3, pp. 146–162, 1954.

[46] T. Hastie and R. Tibshirani, "Classification by pairwise coupling," in *Advances in Neural Information Processing Systems*, M. I. Jordan, M. J. Kearns, and S. A. Solla, Eds., vol. 10. MIT Press, 1998.

[47] D. Heckerman, "A tractable inference algorithm for diagnosing multiple diseases," *arXiv preprint arXiv:1304.1511*, 2013.

[48] L. Jiang and M. Zhou, "Generating Chinese couplets using a statistical MT approach," in *Proceedings of the 22nd International Conference on Computational Linguistics-Volume 1*. Association for Computational Linguistics, 2008, pp. 377–384.

[49] S. Keerthi, S. Shevade, C. Bhattacharyya, and K. Murthy, "Improvements to Platt's SMO Algorithm for SVM Classifier Design," *Neural Computation*, vol. 13, no. 3, pp. 637–649, 2001.

[50] G. Klinge, "Indian summer." in *Haiku in English: The First Hundred Years*, J. Kacian, P. Rowland, and A. Burns, Eds. New York—London: W. W. Norton & Company, 2013, p. 85.

[51] R. Kurzweil, "Ray Kurzweils Cybernetic Poet," *Kurzweil CyberArt Technologies*, 1999.

[52] M. Kusner, Y. Sun, N. Kolkin, and K. Q. Weinberger, "From word embeddings to document distances," in *Proceedings of the 32nd International Conference on Machine Learning (ICML-15)*, 2015, pp. 957–966.

[53] O. Levy and Y. Goldberg, "Neural word embedding as implicit matrix factorization," in *Advances in Neural Information Processing Systems*, 2014, pp. 2177–2185.

[54] T. Lutz, *Über ein Programm zur Erzeugung stochastisch-logistischer Texte*, 1960.

[55] C. D. Manning, "Part-of-speech tagging from 97% to 100%: is it time for some linguistics?" in *Computational Linguistics and Intelligent Text Processing*. Springer, 2011, pp. 171–189.

[56] C. D. Manning, M. Surdeanu, J. Bauer, J. R. Finkel, S. Bethard, and D. McClosky, "The Stanford CoreNLP Natural Language Processing Toolkit." in *ACL (System Demonstrations)*, 2014, pp. 55–60.

[57] H. Manurung, "An evolutionary algorithm approach to poetry generation," 2004.

[58] H. M. Manurung, "Chart generation of rhythm patterned text," in *Proc. of the First International Workshop on Literature in Cognition and Computers*, 1999.

[59] R. Manurung, G. Ritchie, H. Pain, A. Waller, D. O'Mara, and R. Black, "The construction of a pun generator for language skills development," *Applied Artificial Intelligence*, vol. 22, no. 9, pp. 841–869, 2008.

[60] M. P. Marcus, M. A. Marcinkiewicz, and B. Santorini, "Building a large annotated corpus of English: The Penn Treebank," *Computational linguistics*, vol. 19, no. 2, pp. 313–330, 1993.

[61] P. McCorduck, *Aaron's code: meta-art, artificial intelligence, and the work of Harold Cohen*. Macmillan, 1991.

[62] G. Méndez, P. Gervás, and C. León, "On the use of character affinities for story plot generation," in *Knowledge, Information and Creativity Support Systems*. Springer, 2016, pp. 211–225.

[63] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *arXiv preprint arXiv:1301.3781*, 2013.

[64] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in neural information processing systems*, 2013, pp. 3111–3119.

[65] G. Miller and C. Fellbaum, "WordNet: An electronic lexical database," 1998.

[66] A. Mnih and G. E. Hinton, "A scalable hierarchical distributed language model," in *Advances in neural information processing systems*, 2009, pp. 1081–1088.

[67] S. S. Montfort, N., "Sea and spar between," *http://nickm.com/montfort_strickland/sea_and_spar_between/*, 2010.

[68] A. K. Nassirtoussi, S. Aghabozorgi, T. Y. Wah, and D. C. L. Ngo, "Text mining for market prediction: A systematic review," *Expert Systems with Applications*, vol. 41, no. 16, pp. 7653–7670, 2014.

[69] D. L. Nelson, C. L. McEvoy, and S. Dennis, "What is free association and what does it measure?" *Memory & cognition*, vol. 28, no. 6, pp. 887–899, 2000.

[70] D. L. Nelson, C. L. McEvoy, and T. A. Schreiber, "The University of South Florida free association, rhyme, and word fragment norms," *Behavior Research Methods, Instruments, & Computers*, vol. 36, no. 3, pp. 402–407, 2004.

[71] Y. Netzer, D. Gabay, Y. Goldberg, and M. Elhadad, "Gaiku: Generating haiku with word associations norms," in *Proceedings of the Workshop on Computational Approaches to Linguistic Creativity*. Association for Computational Linguistics, 2009, pp. 32–39.

[72] F. Pachet, "The continuator: Musical interaction with style," *Journal of New Music Research*, vol. 32, no. 3, pp. 333–341, 2003.

[73] D. S. Palermo and J. J. Jenkins, "Word association norms: Grade school through college." 1964.

[74] Z. Pan, J. Polden, N. Larkin, S. Van Duin, and J. Norrish, "Recent progress on programming methods for industrial robots," *Robotics and Computer-Integrated Manufacturing*, vol. 28, no. 2, pp. 87–94, 2012.

[75] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu, "BLEU: a method for automatic evaluation of machine translation," in *Proceedings of the 40th annual meeting on association for computational linguistics*. Association for Computational Linguistics, 2002, pp. 311–318.

[76] D. B. Paul and J. M. Baker, "The design for the Wall Street Journal-based CSR corpus," in *Proceedings of the workshop on Speech and Natural Language*. Association for Computational Linguistics, 1992, pp. 357–362.

[77] K. Pearson, "X. on the criterion that a given system of deviations from the probable in the case of a correlated system of variables is such that it can be reasonably supposed to have arisen from random sampling," *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, vol. 50, no. 302, pp. 157–175, 1900.

[78] A. Pease, S. Colton, R. Ramezani, J. Charnley, and K. Reed, "A discussion on serendipity in creative systems," in *Proceedings of the Fourth International Conference on Computational Creativity*, 2013, p. 64.

[79] J. Pennington, R. Socher, and C. D. Manning, "Glove: Global vectors for word representation." in *EMNLP*, vol. 14, 2014, pp. 1532–1543.

[80] B. Perozzi, R. Al-Rfou, and S. Skiena, "Deepwalk: Online learning of social representations," in *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2014, pp. 701–710.

[81] F. Pinel and L. R. Varshney, "Computational creativity for culinary recipes," in *CHI'14 Extended Abstracts on Human Factors in Computing Systems*. Association for Computing Machinery, 2014, pp. 439–442.

[82] J. Platt, "Fast Training of Support Vector Machines using Sequential Minimal Optimization," in *Advances in Kernel Methods - Support Vector Learning*, B. Schoelkopf, C. Burges, and A. Smola, Eds. MIT Press, 1998. [Online]. Available: http://research.microsoft.com/~jplatt/smo.html

[83] Poetry Foundation, http://www.poetryfoundation.org, 2016.

[84] Pre-trained gigaword corpus, http://nlp.stanford.edu/data/glove.6B.zip, 2016.

[85] L. R. Rabiner and B.-H. Juang, "An introduction to Hidden Markov Models," *ASSP Magazine, IEEE*, vol. 3, no. 1, pp. 4–16, 1986.

[86] T. Raworth, "now the melody," in *Haiku in English: The First Hundred Years*, J. Kacian, P. Rowland, and A. Burns, Eds. New York—London: W. W. Norton & Company, 2013, p. 35.

[87] Q. Raymond, *Cent mille milliards de poèmes*. Gallimard, Paris, 1961.

[88] P. Ribeiro, F. C. Pereira, M. Ferrand, A. Cardoso, and P. de Marrocos, "Case-based melody generation with MuzaCazUza," in *Proceedings of the AISB01 Symposium on Artificial Intelligence and Creativity in Arts and Science*. Citeseer, 2001, pp. 67–74.

[89] G. Riley *et al.*, "CLIPS: A tool for building expert systems," *http://www.jsc.nasa.gov/clips/CLIPS.html*, 1999.

[90] G. Salton, "The SMART retrieval system : experiments in automatic document processing," 1971.

[91] G. Salton, A. Wong, and C.-S. Yang, "A vector space model for automatic indexing," *Communications of the ACM*, vol. 18, no. 11, pp. 613–620, 1975.

[92] J. Shapiro and L. Rucker, "Can poetry make better doctors? Teaching the humanities and arts to medical students and residents at the University of California, Irvine, College of Medicine," *Academic Medicine*, vol. 78, no. 10, pp. 953–957, 2003.

[93] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. van den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot *et al.*, "Mastering the game of Go with deep neural networks and tree search," *Nature*, vol. 529, no. 7587, pp. 484–489, 2016.

[94] V. Sorge, S. Colton, R. McCasland, and A. Meier, "Classification results in quasigroup and loop theory via a combination of automated reasoning tools," *Commentationes Mathematicae Universitatis Carolinae*, vol. 49, no. 2, pp. 319–340, 2008.

[95] R. Spiess, "old posts and old wire," in *108 Poems to Cultivate Awareness and Open Your Heart*, P. Donegan, Ed. Boston & London: Shambhala, 2008, p. 159.

[96] M. Steinbach, G. Karypis, V. Kumar *et al.*, "A comparison of document clustering techniques," in *KDD workshop on text mining*, vol. 400, no. 1. Boston, 2000, pp. 525–526.

[97] W. Swist, "far into twilight," in *The Haiku Anthology*, C. van den Heuvel, Ed. New York—London: W. W. Norton & Company, 2000, p. 221.

[98] Twitter, https://about.twitter.com/company, 2016, accessed: 2016-03-30.

[99] C. van den Heuvel, in *The Haiku Anthology*. New York—London: W. W. Norton & Company, 2000.

[100] X. Wan and Y. Peng, "The earth mover's distance as a semantic measure for document similarity," in *Proceedings of the 14th ACM international conference on Information and knowledge management*. Association for Computing Machinery, 2005, pp. 301–302.

[101] R. L. Weide, "The CMU pronouncing dictionary," http://www.speech.cs.cmu.edu/cgi-bin/cmudict, 1998.

[102] M. White, T. Korelsky, C. Cardie, V. Ng, D. Pierce, and K. Wagstaff, "Multidocument summarization via information extraction," in *Proceedings of the first international conference on Human language technology research*. Association for Computational Linguistics, 2001, pp. 1–7.

[103] M. Wick and B. Vatant, "The geonames geographical database," *Available from World Wide Web: http://geonames. org*, 2012.

[104] D. Winer, "Review of ontology based storytelling devices," in *Language, Culture, Computation. Computing of the Humanities, Law, and Narratives*. Springer, 2014, pp. 394–405.

[105] M. T. Wong, A. H. W. Chun, Q. Li, S. Chen, and A. Xu, "Automatic haiku generation using VSM," in *WSEAS International Conference. Proceedings. Mathematics and Computers in Science and Engineering*, no. 7. World Scientific and Engineering Academy and Society, 2008.

[106] Zack, "Turing test: Passed, using computer-generated poetry," https://rpiai.wordpress.com/2015/01/24/turing-test-passed-using-computer-generated-poetry, 2015.

[107] C. Zhai and J. Lafferty, "A study of smoothing methods for language models applied to ad hoc information retrieval," in *Proceedings of the 24th Annual International Association for Computing Machinery: SIGIR conference on Research and development in information retrieval*. Association for Computing Machinery, 2001, pp. 334–342.

APPENDIX A

IRB APPROVAL

ASU Knowledge Enterprise Development

EXEMPTION GRANTED

Rida Bazzi
Computing, Informatics and Decision Systems Engineering, School of (CIDSE)
480/965-2796
bazzi@asu.edu

Dear Rida Bazzi:
On 3/16/2016 the ASU IRB reviewed the following protocol:

Type of Review:	Initial Study
Title:	A user study to evaluate an automated fixed verse generation technique
Investigator:	Rida Bazzi
IRB ID:	STUDY00004055
Funding:	None
Grant Title:	None
Grant ID:	None
Documents Reviewed:	• Questionnaire Experiment Group, Category: Measures (Survey questions/Interview questions /interview guides/focus group questions); • Questionnaire Control Group, Category: Measures (Survey questions/Interview questions /interview guides/focus group questions); • HRP-502a Consent Document, Category: Consent Form; • HRP-503a Protocol, Category: IRB Protocol;

The IRB determined that the protocol is considered exempt pursuant to Federal Regulations 45CFR46 (2) Tests, surveys, interviews, or observation on 3/16/2016.

In conducting this protocol you are required to follow the requirements listed in the INVESTIGATOR MANUAL (HRP-103).

Sincerely,
IRB Administrator

cc: Arjun Magge Ranganatha

APPENDIX B

CONSENT FORM FOR THE USER STUDY

A USER STUDY TO EVALUATE AN AUTOMATED FIXED VERSE GENERATION TECHNIQUE

I am a student at the School of Computing, Informatics and Decision Systems Engineering at Arizona State University and as part of my thesis, I am conducting a user study to evaluate automatic fixed verse generation techniques.

I am inviting your participation, which will involve you to evaluate a small collection of poems across six qualities expected in a poem. The total time that will be taken to finish this survey is around 15 minutes. You have the right not to answer any question, and to stop your participation at any time.

Your participation in this study is voluntary. If you choose not to participate or to withdraw from the study at any time, there will be no penalty. However, to participate, you must be at least 18 years of age. There are no foreseeable risks or discomforts to your participation.

Your responses will be completely anonymous. You are not required to provide any information with which your identity can be traced back, i.e. information like your name, ASU ID etc. The results of this study may be used in reports, presentations, or publications and your name will not be used.

There will be no audio or video recording of the study. As mentioned earlier, only a survey needs to be filled by the participants. If you have any questions concerning the research study, please call me at 480-965-2796 or send me an email at bazzi@asu.edu. If you have any questions about your rights as a subject/participant in this research, or if you feel you have been placed at risk, you can contact the Chair of the Human Subjects Institutional Review Board, through the ASU Office of Research Integrity and Assurance, at (480) 965-6788.

If you fill up the survey questionnaire, it will be considered as your consent to participate in the study. Your participation is highly appreciated.

Thank you,
Arjun Magge,
Graduate Student,
Computer Science, CIDSE,
Arizona State University.

APPENDIX C

QUESTIONNAIRES FOR THE USER STUDY

A USER STUDY TO EVALUATE AN AUTOMATED FIXED VERSE GENERATION TECHNIQUE

Demographic Information

[NOTE : Section I is common both the control and experimental group and stayed unchanged across the groups.]

Section I

This section consists of questions that have been designed to collect the information necessary to determine your expertise and familiarity in the area of English literature and poetry.

Please answer the following questions about yourself.

1. What is your proficiency in English?

   ○ Beginner

   ○ Conversational

   ○ Fluent

   ○ Native

2. Are you currently (or formerly) a student of English literature or poetry at a University?

   ○ Yes

   ○ No

3. Do you actively seek out poetry in magazines, books, blogs and publications for reading?

   ○ Yes

   ○ No

   ○ Other :_____

4. On an average, how many poems do you read in a month?

   ○ _____

Control Group

Section II

This section has a collection of six short poems that need evaluation. The collection contains three poems composed by humans and three poems generated by a computer. Each poem is accompanied by five questions designed to evaluate the quality of the poem that require your response.

[NOTE : For the sake of conciseness we include the format for only one human composed poem and one machine generated poem. The actual study for the control group would contain three human composed poems followed by three machine generated poems.]

Please read the following HUMAN COMPOSED poem and answer the questions below:

Human Composed Poem #1

pine shade
the wooden bench
worn smooth

1. Did the poem include any of the necessary poetic devices: rhyme, rhythm, alliteration, personification etc. that raise the poetry beyond prose?

   ○ Yes
   ○ No
   ○ Don't know

2. Does the text evoke emotions or mental images?

   ○ Yes
   ○ No
   ○ Don't know

3. Were there clichés or overused imagery that weaken the conveyance of meaning?

   ○ Yes
   ○ No
   ○ Don't know

4. Were there distortions of word order that seemed artificial?

   ○ Yes
   ○ No
   ○ Don't know

5. How artfully has the poem been rendered? (encircle)

Poorly 1 2 3 4 5 Efficiently

Comments about the poem (optional)

Please read the following MACHINE GENERATED poem and answer the questions below:

Machine Generated Poem #1

sour shear
the wistful crowd
swum seaward

1. Did the poem include any of the necessary poetic devices: rhyme, rhythm, alliteration, personification etc. that raise the poetry beyond prose?

   ○ Yes
   ○ No
   ○ Don't know

2. Does the text evoke emotions or mental images?

   ○ Yes
   ○ No
   ○ Don't know

3. Were there clichés or overused imagery that weaken the conveyance of meaning?

   ○ Yes
   ○ No
   ○ Don't know

4. Were there distortions of word order that seemed artificial?

   ○ Yes
   ○ No
   ○ Don't know

5. How artfully has the poem been rendered? (encircle)

Poorly 1 2 3 4 5 Efficiently

Comments about the poem (optional)

Experimental Group

Section II

This section has a collection of six short poems that need evaluation. The collection may contain anywhere from zero to six poems written by humans and the remaining generated by a computer. Each poem is accompanied by six questions designed to evaluate the quality of the poem that require your response.

[NOTE : For the sake of conciseness we include the format for only poem. The actual study for the experimental group would contain a total of six poems.]

Please read the following poem and answer the questions below:

Poem #1

pine shade
the wooden bench
worn smooth

1. Did the poem include any of the necessary poetic devices: rhyme, rhythm, alliteration, personification etc. that raise the poetry beyond prose?

   ○ Yes
   ○ No
   ○ Don't know

2. Does the text evoke emotions or mental images?

   ○ Yes
   ○ No
   ○ Don't know

3. Were there clichés or overused imagery that weaken the conveyance of meaning?

   ○ Yes
   ○ No
   ○ Don't know

4. Were there distortions of word order that seemed artificial?

   ○ Yes
   ○ No
   ○ Don't know

5. How artfully has the poem been rendered? (encircle)

   Poorly 1 2 3 4 5 Efficiently

6. Who is most likely to have composed the above poem?

   ○ A Human
   ○ A Computer
   ○ Don't know

Comments about the poem (optional)

APPENDIX D

POEMS USED FOR THE STUDY

Human Composed Poems

All URLs accessed : 2016-03-30

now the melody
in the pattern of shadows
one shadow behind

- [86]

old posts and old wire
holding wild grape vines holding
old posts and old wire

- [95]

far into twilight
milkweed crosses the meadow
the evening star

- [97]

Indian summer.
Even a small affection
has its urgency.

- [50]

Winter's Midnight muse
Spring's forgotten library
Come You're far from me.

- @AmyBillone
https://twitter.com/AmyBillone/status/711039898665390081

Winter end sunny.
Spring start cold, rainy, lively.
I dance, sing, write, tweet.

- @serenespaceskl
https://twitter.com/serenespaceskl/status/711945392212959232

Clueless Mr. Jones
Runs Fine Lines Between Grounds Coarse
Sounds Always Brew True

- @dalien77
https://twitter.com/dalien77/status/712775916334620672

Blow, bubbles floating
Plastic mirrors undulate
Breaking your mind up

- @CarnivaleDTriam
https://twitter.com/CarnivaleDTriam/status/712808309137543168

There was a young lady of Cork,
Whose Pa made a fortune in pork;
He bought for his daughter
A tutor who taught her
To balance green peas on her fork.

- Anonymous
http://irish.spike-jamie.com/limericks.html

There was a young lady named Perkins,
Who just simply doted on gherkins.
In spite of advice,
She ate so much spice,
That she pickled her internal workins'.

- Anonymous
http://www.oldfashionedamericanhumor.com/limericks.html

It snowed and snowed on the day I was born,
And for that reason I love falling snow;
In dreams, I walk the frozen woods till morn;
And from my pen the words just drip and flow.

- Broken Wings
http://www.poetrysoup.com/poem/it_was_snowing_771929

Love firmly planted builds a broader view;
rose tinted glasses traded for the clear,
to open up the narrow paths, breakthrough
to fuller scenes that blinded love may fear.

- Sandra M. Haight
http://www.poetrysoup.com/poem/love_is_blind_771510

Machine Generated Poems

Liquid ambience
Vanilla Chocolate Milk Cream.
Soft, bottled juices.

distinctive color
yellow Philadelphia
any blue your black

seasonal harvests
plentiful abundant crops
natural winter

branding parody
characters love forever
wonder inspired

True reality,
I might always mind your kind.
Beauty, characters.

winter's snow around
rainy days, coming age
before starting down.

Unique example
rather beyond creating.
Way, creates my life.

Soft, eating some eggs
Andromeda tomato.
Sunscreen melamine.

The stripes of blue, black, yellow, red and pink;
Their purple shaped with colors and green shades.
Except arm's white and eyes they may rethink
A gray of dark and light; with no bright blades.

And from a life I was even, and kind,
comin' to own well things, and own my means;
I would backstitch in making and be signed,
Doin' the years, an instance of own scenes.

There was a fat sugar of Lemon
Who cooked as she dried on a chicken;
They consumed from the milk
With the butter in silk,
And the juice on the cream of the poison.

There once was a blue striped Violet,
A black, analyzable granite;
After wearing his white,
Then he wore his own light
That red wasn't green, he was desperate.

APPENDIX E

HASHTAGS USED FOR MONITORING TWITTER

#haiku #haikuchallenge #senryu #botaiku #micropoem
#micropoetry #mpy #MadVerse #fsmpy #vss
#poem #poetry #fieryverse #8words #inkedverse
#MSpoetry #SLPoetry #POMwords #DsubVerse #ashverse
#3lines #WyldeVerse #RavensVeil #orjay #bibtp
#WrittenRiver #tanka #5lines #madverse #writtenriver
#mspoetry #pomwords #dsubverse #slpoetry #ravensveil
#prompt #microprompt #sixwords #asymlife #wyldeverse
#verse #sensewrds #poemtrail #sedserio #lqw
#museinlove #soulwords #6words #ntitle #highonpoetry
#carnalverse #poetweet #drugverse